Coded Aperture Imaging

Manuel Martinello

School of Engineering and Physical Sciences

Heriot-Watt University

A thesis submitted for the degree of

PhilosophiæDoctor (PhD)

May 2012

1. Reviewer: Prof. Richard Staunton (University of Warwick)

2. Reviewer: Prof. Yvan Petillot (Heriot-Watt University)

Day of the defence: 20 April 2012

Signature from head of PhD committee:

Abstract

This thesis studies the coded aperture camera, a device consisting of a conventional camera with a modified aperture mask, that enables the recovery of both depth map and all-in-focus image from a single 2D input image. Key contributions of this work are the modeling of the statistics of natural images and the design of efficient blur identification methods in a Bayesian framework. Two cases are distinguished: 1) when the aperture can be decomposed in a small set of identical holes, and 2) when the aperture has a more general configuration. In the first case, the formulation of the problem incorporates priors about the statistical variation of the texture to avoid ambiguities in the solution. This allows to bypass the recovery of the sharp image and concentrate only on estimating depth. In the second case, the depth reconstruction is addressed via convolutions with a bank of linear filters. Key advantages over competing methods are the higher numerical stability and the ability to deal with large blur. The all-in-focus image can then be recovered by using a deconvolution step with the estimated depth map. Furthermore, for the purpose of depth estimation alone, the proposed algorithm does not require information about the mask in use. The comparison with existing algorithms in the literature shows that the proposed methods achieve state-of-the-art performance. This solution is also extended for the first time to images affected by both defocus and motion blur and, finally, to video sequences with moving and deformable objects.

To Sobia, my strength ...

Acknowledgements

Apart from my effort, the success of this project depends largely on the encouragement and guidelines of many others. I take this opportunity to express my gratitude to the people who have been instrumental in the successful completion of this work.

I am very grateful to my supervisor Paolo Favaro for his thorough supervision and the stream of ideas that kept me occupied. Our discussions over the years have consistently been challenging and creative, and most of the ideas of this thesis were born from this interaction.

I wish to express my gratitude to Tom Bishop, who introduced me in the "world of photography". During my PhD he has always been a very helpful support when I did not know where to bang my head. Part of his contribution to this work is present in Chapter 5.

I would also like to acknowledge some colleagues and friends - Riccardo, Gerard, Jonny, Qingxu, Thoma, Daniele, Calum, Eleonora - who contributed to make the lab a nicer and enjoyable environment. After this thesis I can be back to the 'social' life.

Many thanks to the group MaPaCaPoGia (Palme, Cam, Pozz, and Jack), from whom I learned the real value of "friendship". I especially thank Camillo for having listened all my stories during the time spent in the flat.

The work in this thesis was funded by the School of Engineering and Physical Sciences at Heriot-Watt University and by Selex Galileo.

I reserve a few more words for some very important persons of my life.

I will never say “Thank You” enough to Sobia for giving me the strength and the help I needed when I needed, and for having read my thesis ’full of funny symbols’.

A special “Grazie” goes to my parents, Ivana and Elia, and my sister Vanessa, for the support, the encourage, and for all the hope they have on me. I hope I can give something back soon.

The last thought goes to two very important people who passed away during my PhD, my grandma Assunta and my grandpa Silvio: I am sure you would be very proud of me!

Contents

List of Figures

List of Tables

List of Publications

Peer reviewed conferences / journals

1. M. Martinello, T. E. Bishop, and P. Favaro. A bayesian approach to shape from coded aperture. IEEE International Conference of Image Processing, Sep 2010.

2. M.Martinello and P. Favaro. Single image blind deconvolution with higher-order texture statistics. Video Processing and Computational Video, LNCS 7082, pp. 124-151, Springer-Verlag, 2011.

3. M. Martinello and P. Favaro. Fragmented aperture imaging for motion and defocus deblurring. IEEE International Conference of Image Processing, Sep 2011.

4. M. Martinello and P. Favaro. Depth estimation from a video sequence with moving and deformable objects. IET Image Processing, Jul 2012.

Demos

1. M. Martinello and P. Favaro, Coded Aperture Videography, IEEE Computer Vision and Pattern Recognition, Jun 2011.

Awards

1. M. Martinello, 3D information from one single 2D image, Set for Britain 2011, House of Commons, Westminster, London, Mar 2011. [The Parliamentary and Scientific Committee silver award - engineering research category]

This page has been left intentionally blank.

Chapter 1

Introduction

The brick walls are not there to keep us out.
The brick walls are there to give us a chance
to show how badly we want something.
Because the brick walls are there to stop
the people who don't want it badly enough.
Randy Pausch [1960-2008]

This thesis presents an investigation on 3D scene reconstruction from 2D images and videos. The ability to reconstruct a 3D scene is important for cultural heritage preservation, computer games, movies, and city modelling, but it is fundamental for autonomous navigation. Applications for autonomous navigation and assisted autonomous navigation include investigating highly-unsafe areas (*e.g.*, fires or radioactive areas), checking line integrity of underwater pipelines, and assisting drivers to avoid collisions.

One of the most crucial tasks in autonomous navigation is to obtain the 3D information of the environment to either avoid or engage with 3D objects in front of the vehicle. There are several approaches to capture 3D information. The most common techniques make use of two (stereo) or more cameras to capture different viewpoints of the scene. Other methods consider systems with active sensors, which provide their

own energy source for illumination. These systems emit radiation toward the target of interest, and then measure the radiation reflected by the target.

The widespread use of cameras in mobile phones has generated a strong interest in reducing the overall camera size. However, smaller imaging devices would also be beneficial to autonomous systems as they would leave more space to batteries and weigh less. In this context, multiple camera systems are not desirable. Moreover, multiple cameras also require additional electronics for synchronization. An active system may solve the problem of size, but in this case the main drawback is its limited autonomy: The sensors, in fact, require the generation of a fairly large amount of energy to adequately illuminate targets. Moreover, active systems are more expensive than passive systems and do not solve the problem entirely: For example, *kinect* (a depth camera recently introduced [99]) works very well indoor, but is not reliable when used outdoor [76].

Therefore there is a strong interest in studying low-cost systems that can be scaled and that have limited power requirements. In this context, this thesis looks at passive approaches to extracting 3D information of a scene from a single camera. Furthermore, particular attention is given to extracting such information based on a single image since objects in the scene can move independently of each other. However this is an extremely challenging problem: Images are the result of a projection of the 3D scene onto two dimensions. Thus, one dimension is somehow lost in the process. Typically, one associates the lost dimension to *depth*, i.e., the distance of an object from the camera. However, depth is not necessarily lost forever in an image. Indeed, the next section will illustrate that cameras may be capable of encoding depth in an image by trading off other visual information.

1.1 Depth from a Single Image

When capturing an image, objects at the focal plane appear sharp while objects located away from the focal plane appear out-of-focus. For example, Figure 1.1(a) shows three

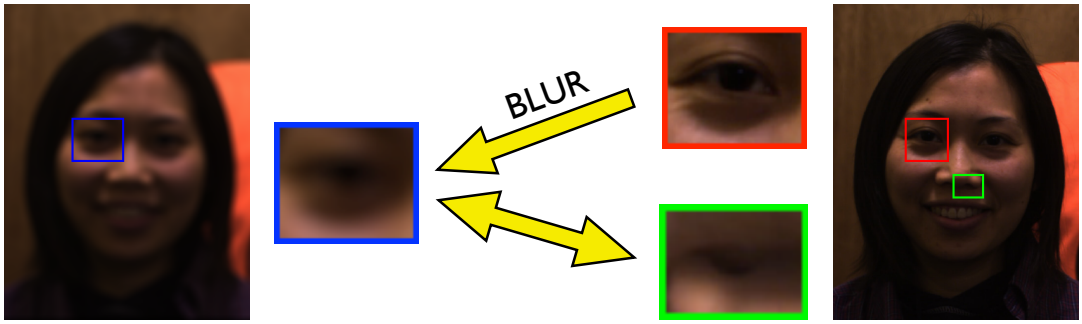
(a) (b)

Figure 1.1: In-focus and out-of-focus in a picture. (a) Picture of three persons from [50]: when the woman is brought into focus, the two men behind her appear out-of-focus. (b) The original images of the faces of the two men, before being blurred by the camera.

individuals and only one of them is in focus. It is quite challenging to recognize the two persons in the background. To recover their faces when in-focus (as displayed in Figure 1.1(b)) starting from the blurred image in Figure 1.1(a), one must know how much blur has been added by the camera when the picture was captured, and then remove it. The former task is usually termed *blur estimation*, while the latter is defined as *image deblurring*.

Notice that the amount of blur depends on the distance from the focal plane. For example, in Figure 1.1(a) the third individual is more blurred than the second one. Thus if we are able to identify the amount of blur of an object in the image, we have some information about its distance from the camera. For this reason the blur estimation task is equivalent to a *depth estimation* task.

However blur estimation is made challenging by the difficulty of distinguishing different amounts of blur. In the next section these challenges will be discussed and an approach to address them outlined.



(a) Out-of-focus (b) Very likely blur ambiguity (c) In-focus

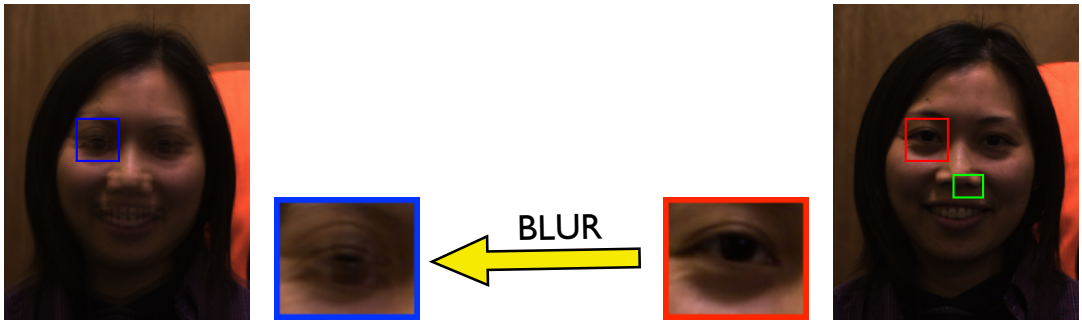
Figure 1.2: Out-of-focus effect in a conventional camera. Ambiguity when trying to identify the amount of blur: If we extract a patch from the out-of-focus image (a), this can be considered to be either a blurred version of the right eye (red box) or a sharp patch containing the left nostril (green box).

1.2 Disadvantages of Conventional Cameras

Looking at a picture taken with a camera, there are textures that are blurred and texture that are in-focus but "seem" blurred. For example, consider the out-of-focus picture of the woman displayed in Figure 1.2(a). The blurred patch of her right eye can be mistaken for the in-focus patch of her left nostril. This ambiguity comes from the fact that conventional cameras generate out-of-focus images whose patches may be very similar to other natural textures; this makes blur identification harder to solve. To reduce the ambiguity one can modify the blur pattern of a conventional camera so that it generates an out-of-focus image that is as different as possible from a natural. This new device is called a *coded aperture camera*. An example of its output is the image Figure 1.3(a). In this case the blurred patch of the right eye has a unique solution (see Figure 1.3(b)). In fact, coded aperture cameras create patterns in the out-of-focus image that are very different from natural texture, and therefore easier to identify.

1.3 From Autostereograms to the Coded Aperture Camera

The image obtained from a coded aperture device is based on the same principle that an *autostereogram* uses to encode the depth information. An autostereogram is a man-



(a) Out-of-focus (b) Unlikely blur ambiguity (c) In-focus

Figure 1.3: Out-of-focus effect in a coded aperture camera. When using a coded aperture camera (in this case, the aperture mask is composed by two vertically-displaced holes), the blur ambiguity is reduced. The patch from the out-of-focus image contains a blur pattern which is easier to distinguish from the natural texture.

made single image designed to create the visual illusion of a 3D scene from a 2D image in the human brain. The simplest type of autostereogram, like the one shown in Figure 1.4, consists of horizontally repeating patterns. The distance between one repetition and the other gives the 3D information.

In an autostereogram the texture is entirely artificial in order to keep as much depth information as possible in the image. In our case we cannot modify the texture of the scene, so there is a trade-off between texture resolution and depth information.

A better understanding of how this technique can be implemented and the changes needed in the camera system is required. For this purpose one can analyze how the brain perceives distances through our eyes and how this is used to "see" the 3D scene in an autostereogram.

1.3.1 3D Perception in the Human Brain

The eye can be compared to a photographic camera. It has an adjustable pupil which can open (or close) to allow more (or less) light to enter the eye. As with any camera, the light rays entering through the pupil (aperture in a camera) need to be focused on a single point on the retina in order to produce a sharp image. The eye achieves this goal by adjusting a lens behind the cornea to refract light appropriately (Figure 1.5(a)).

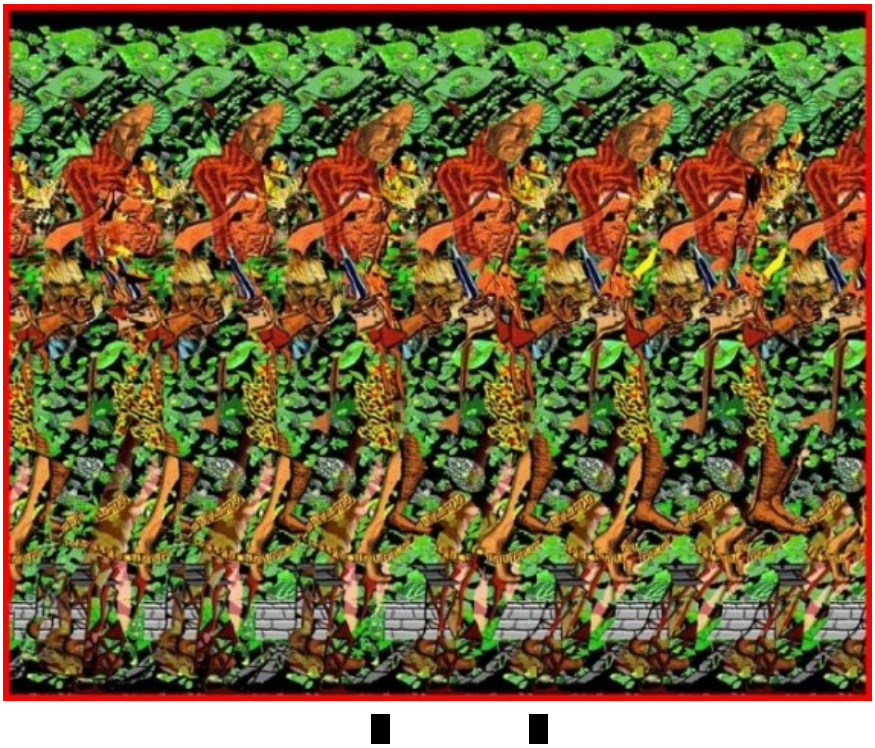
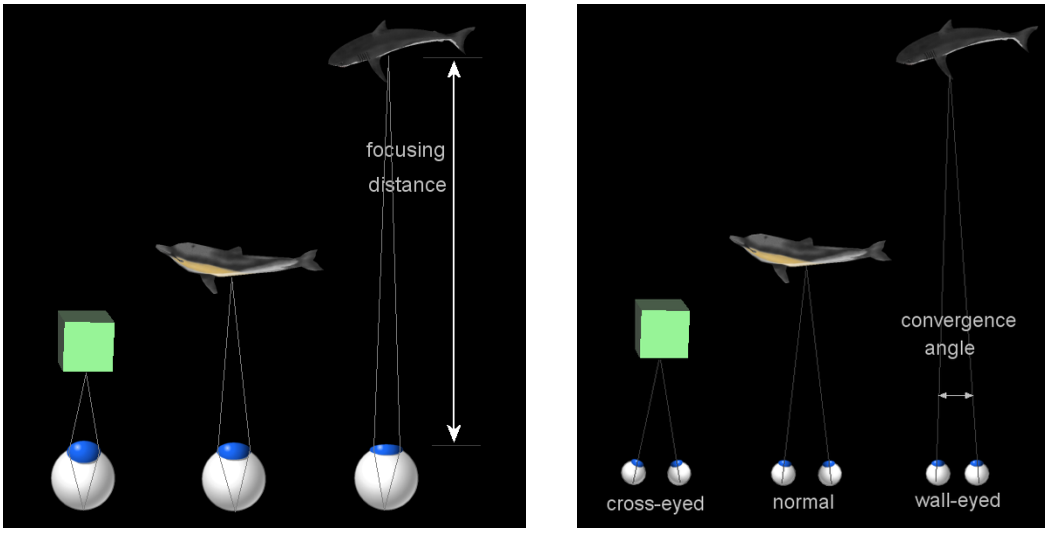


Figure 1.4: Autostereogram. What do you see in the image? To facilitate the 3D perception, look at the black rectangles. Cross your eyes until you see a third black rectangle between them and then focus on it. Once you can clearly see the third rectangle, move your eyes on the image, but make sure you do not change the focus, and observe the image. Move the head slightly sideways to perceive the depth (the depth map is shown in Figure 1.7). Image taken from [2].

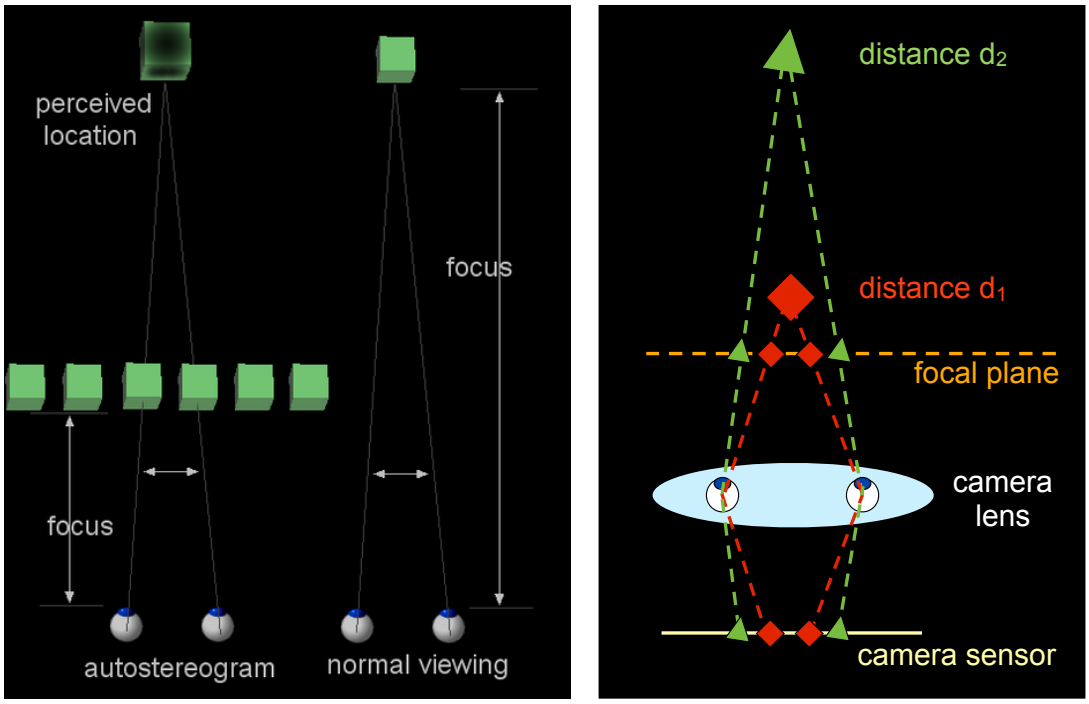
When a person stares at an object, the two eyeballs rotate sideways to point to the object of interest, so that it appears at the center of the image formed on each eye's retina. When looking at a nearby object, the two eyeballs rotate towards each other so that their eyesight can converge on the object. This is referred to as *cross-eyed* viewing. In contrast, to see a distant object the two eyeballs diverge to become almost parallel to each other. This is known as *wall-eyed* viewing (also known informally as parallel-viewing), where the convergence angle is much smaller than that in a cross-eyed viewing [93]. Figure 1.5(b) shows how the eye convergence varies depending on the position of the object of interest. In particular, the convergence angle allows the brain to calculate distance of objects relative to the point of convergence.



(a) Focus (b) Convergence

Figure 1.5: Difference between focus and convergence of the eyes. (a) Each eye adjusts its internal lens to get a clear, focused image; (b) The two eyes converge to point to the same object. Images taken from [93].

The eyes normally focus and converge at the same distance: When looking at a distant object, the brain automatically flattens the eye lenses and rotates the two eyeballs for wall-eyed viewing. It is possible to train the brain to separate the focus point from the convergence point. This decoupling has no useful purpose in everyday life, since it prevents the brain from interpreting objects in a coherent manner. However, it is crucial in order to see an autostereogram, such as the one shown in Figure 1.4. An observer can construct a 3D interpretation in his or her perception by matching picture elements along horizontal lines from the image plane. Figure 1.6(a) demonstrates how this technique works in more detail. By focusing the lenses on a nearby autostereogram where patterns are repeated, and by converging the eyeballs at a distant point behind the autostereogram image, one can trick the brain into seeing 3D images. If the patterns received by the two eyes are similar enough, the brain will consider these two patterns a match and treat them as coming from the same imaginary object located at the convergence point of the eyes.



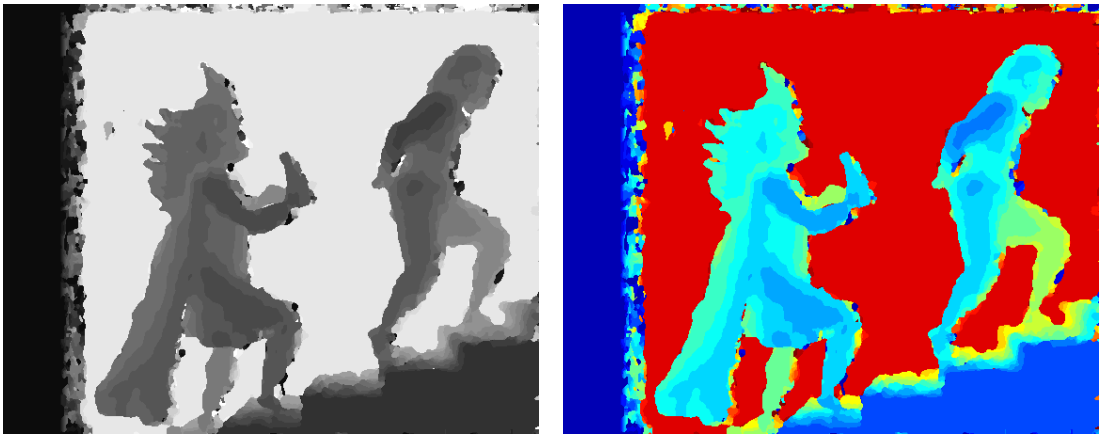
(a) Technique to "see" an autostreogram (b) Coded aperture camera model

Figure 1.6: From autostereograms to coded aperture camera. (a) Decoupling focus from convergence tricks the brain into seeing 3D images in a 2D autostereogram. (b) Model of the simplest coded aperture camera, where the mask is composed by just two holes in the lens, similar to human eyes.

1.3.2 From Vision to Hardware

This technique can be easily applied to our conventional camera, as shown in Figure 1.6(b). Instead of using the whole lens when capturing an image, consider only the light going through two openings, which correspond to the pupils of the human eyes. The eye lens is represented by the main camera lens, which is now in common between the two openings.

When reading the autostereograms (Figure 1.6(a)), start from the image and project the double pattern into the scene to perceive the object at the eye convergence point. When capturing a picture with our modified camera (Figure 1.6(b)), start from the point of convergence, which represents the location of the object in the scene, and record its double image in our camera sensor. The image in the sensor will be the



(a) Depth map in grayscale (b) Depth map with colors

Figure 1.7: Example of depth map. The depth map of the autostereogram in Figure 1.4 is shown in (a) grayscale and (b) with colors. Depth map is a one-channel image whose values indicate the distance from the camera.

same (a flipped version, to be precise) of the one that is formed at the focal plane of the main lens. With this setting all the items placed further than the focal plane will have a double image, and the distance between the two projections (or repetitions) will depend on the location of the object in the scene.

Thus, by reporting the values of the distance of the repetitions at each pixel of the captured image, one can obtain a one-channel image, known as *depth map*. The value of a pixel in a depth map represents the relative distance from the focal plane, where instead the projections coincide. An example of depth map is displayed in Figure 1.7 in two different formats: grayscale and colored. In Figure 1.7(a) nearer surfaces are darker, while further surfaces are brighter; In Figure 1.7(b) cold colors (blue) indicate areas close to the camera, while hot colors (red) indicate distant areas. This depth map in Figure 1.7 represents the distance of the repetitions of the pattern in the autostereogram of Figure 1.4, and therefore the 3D scene that the observer's eyes should "see".

We have seen the example with two holes, but this idea can be applied with any number of openings in the lens: The difference is that there will be several projections

of the same object in the out-of-focus image, one for each hole that composes the aperture mask. In general, a mask (a piece of cardboard is sufficient) can be built with any binary pattern and placed on the main lens of the camera. Essentially, instead of having the blur created by the whole lens, in a coded aperture camera one can design a specific pattern for the blur so that it is easier to separate from the natural texture when there is an out-of-focus image.

1.4 Contributions of this Thesis

This thesis presents an analysis of what aperture masks are optimal for reconstruction and the corresponding algorithms to obtain it. A key contribution in this approach is the modelling of the statistics of natural images and the design of efficient blur identification methods. Two cases are distinguished: When the aperture can be decomposed in a small set of identical openings (*simple patterns*), and when it is a more general configuration (*general patterns*).

In the first case, the reconstruction of the sharp image is avoided by incorporating image priors about the local space-varying texture statistics in a Bayesian framework. Since the problem is formulated as linear in the unknown sharp image, a closed-form solution can be obtained so that it depends only on the depth map [62].

In the second case, the depth reconstruction is addressed via convolutions with a bank of linear filters. This approach is in contrast to other competing methods based on deconvolution. Key advantages are the higher numerical stability and the ability to deal with large blur. The all-in-focus image can then be recovered by using a deconvolution step with the estimated depth map. Furthermore, for the purpose of depth estimation alone, the proposed algorithm does not require calibration as the bank of filters can be learned directly from blurred images (i.e., one does not need to know the aperture mask). Results on both synthetic and real data are presented and compared to existing algorithms in the literature, showing that the proposed methods achieve state-of-the-art performance, without any user intervention [60]. This approach is also

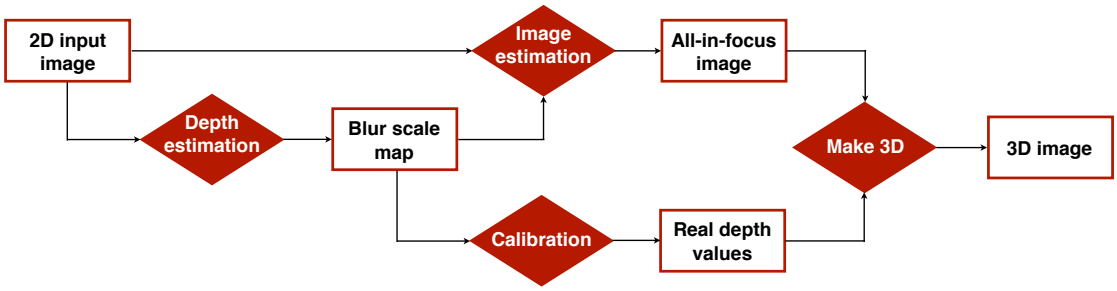


Figure 1.8: Depth and image from a single 2D image. The graph represents the steps which this approach is divided in.

extended for the first time to the other two very challenging situations: 1) an image affected by both defocus and motion blur [59] and 2) a monocular video sequence, when moving and deformable objects are present in the scene [61]. For both cases, successful results are achieved.

Finally, the thesis presents a novel technique to design optical coded apertures, which is based on a geometrical interpretation of blurred images.

1.5 Thesis Structure

After presenting an overview of the most relevant prior work in Chapter 2, some notions of depth from defocus are recalled in Chapter 3 to describe the image formation model of a coded aperture camera.

Chapter 4 analyzes the problem of reconstructing both depth and all-in-focus image from a single coded image, and discusses some previous solutions and their limitations. The approach presented in this thesis can be decomposed in different steps, which are illustrated in Figure 1.8 for the benefit of the reader. The most important steps of this problem are the depth (or blur) estimation and the image deblurring. The former step produces a depth map, also referred to as blur scale map since it is based on a blur scale identification. The values of the map can be easily turned into real depth values by using the calibration procedure described in the last part of Chapter 3. Two novel approaches are proposed to solve the depth estimation step, de-

(a) Input - 2D single coded image (b) Output - 3D image

Figure 1.9: Example of input and output of the proposed technique. (a) a single 2D image of the scene is captured with the coded aperture camera; (b) the proposed approach estimates depth and all-in-focus image, which can be combined together in a 3D image (to be watched with red-cyan glasses).

pending on the pattern of the mask in use: Chapter 5 describes an efficient method for patterns that can be decomposed in a small set of identical opening; Chapter 6 presents a method based on a study of subspaces that can deal with any type of pattern in the aperture mask.

When the depth map has been estimated, the image estimation step can be performed on the coded image, as described in the second part of Chapter 6. Finally, the information from the depth map and the all-in-focus image can be combined together in a 3D image (Figure 1.9), which simulates a stereoscopic effect if watched with red-cyan glasses.

Once successful results are obtained from a single image, the problem is extended to a more generic scenario where objects can move independently. Chapter 7 considers the case when both defocus and motion blur affect a single shot, while Chapter 8 investigates how to adapt the proposed approach to a video sequence, and make use of the information from different frames to improve the quality of the depth maps.

To conclude, a geometric interpretation of blurred images is presented in Chapter 9. Such interpretation enables the design of a coded aperture selection criterion, which is applied to all the patterns previously used in literature.

This page has been left intentionally blank.

Chapter 2

Literature Overview

Experience is what you get when you didn't get what you wanted.
And experience is often the most valuable thing you have to offer.
Randy Pausch [1960-2008]

This chapter provides an overview of the most important related work that has been carried out in the past, highlighting advantages and limitations of previous approaches. A large amount of work has been undertaken to solve the problem of estimating both depth and all-in-focus image from multiple defocused images, but there is a very limited contribution regarding the case when the input is restricted to a single defocus image.

The chapter is divided in two parts: Section 2.1 contains prior work that has been carried out with a conventional camera, while Section 2.2 analyzes work where the aperture of the camera lens has been modified with additional optical elements (Section 2.2.1) or with binary aperture masks (Section 2.2.2).

2.1 Depth from Defocus

The previous chapter has introduced the relationship between defocus blur and distance from the camera. *Depth from defocus* (DFD) is a technique in which the blur at a pixel in an image is used to estimate the distance from the lens to the corresponding

point on an object. The method requires a set of differently focused images acquired from a single view point using a single camera.

The most direct approach to DFD is to formulate an image deblurring problem: This consists on seeking the in-focus image of the scene and the defocus parameters that best reproduce two or more input images acquired at different lens settings. Since this formulation is based on deconvolution (a well-known *ill-posed* problem [5]), and the input data may contain noise, additional smoothness terms are required to regularize the optimization [28, 79]. Considering the image deblurring as a global optimization may results on a very intractable problem. Normally deblurring methods make use of some iterative refinement techniques, such as gradient descent flow [43], EM-like alternating minimization [26, 38], or simulated annealing [7, 79]. These methods have the disadvantages of being sensitive to the initial estimate, and may potentially become trapped in local extrema.

Alternatively, one can factor out the texture of the underlying scene, by estimating the relative defocus between the input images instead. Finally, if the prior knowledge of the scene is strong enough, different defocus hypotheses can be directly evaluated using just a single image.

These methods are divided depending on the number of input images being used: approaches that require multiple images are described in Section 2.1.1, while those using only a single image are reviewed in Section 2.1.2.

2.1.1 Using Multiple Images

MRF-Based Models One simplifying approach to image restoration is to discretize the scene into additive fronto-parallel depth layers. If the layers are modelled as opaque, then every pixel is assigned to a single depth layer, casting depth recovery as a combinatorial assignment problem [38, 79]. This problem can be addressed using a Markov Random Field (MRF) framework [10], based on formulating costs for assigning a depth layer (which corresponds to a discrete defocus label) to each pixel, as well as smoothness costs favouring adjacent pixels with similar depth (or defocus) val-

ues. In [79] the authors formulate a spatially-variant model of defocus (Section 3.2.3) in terms of an MRF, and suggest to optimize the MRF using a simulated annealing procedure, initialized with a classic window-based DFD method.

Differential Defocus Farid realizes an interesting version of differential DFD [24], by using specially designed pairs of optical filters that *directly* measure derivatives with respect to the aperture size or viewpoint. By comparing the image produced with one filter, and the spatial derivative of the image generated with another filter, the authors obtain a scale factor for every point; this can then be related to depth. This method relies on defocus, otherwise the scale factor will be undefined.

Rational Filters Watanabe and Nayar [100] propose the use of rational filters for passive DFD. They consider the amplitude ratio between the difference (M) of the defocused images to their sum (P), and develop a set of broadband filters that model the M/P ratio as a function of depth. They consider a relatively small 7×7 kernel. Although filters are designed in the frequency domain, the depth estimation algorithm is implemented in the spatial domain, resulting in efficient 2D convolutions. However, the main drawback of this technique is the filters design procedure. The authors propose a complicated iterative minimization technique to model the rational filters for any given defocus condition and any texture frequency.

Very recently, the filter design has been simplified by Raj and Staunton [77]. They present a novel procedure that avoids the iterative minimization, and results in filters that are largely insensitive to object texture and model the blur more precisely than [100].

Orthogonal Filters In [27], Favaro and Soatto show how the problem of estimating both depth and all-in-focus image from blurred images, can be performed in two separate steps, without loss of solutions: 1) depth reconstruction first, and then 2) image deblurring, using the estimated depth. In the first part blur kernel can be recovered by using a set of orthogonal filters, which characterize the relative defocus

at particular calibrated depths. The approach is based on a learning procedure where a large number of defocused images are combined to recover an operator that describes a linear subspace (related to a defocus level) and provides invariance to the scene radiance.

2.1.2 Using a Single Image

Recently, an increasing amount of algorithms have proposed to estimate depth and deblurred image from just a single defocused image, captured by an uncalibrated conventional camera: Namboodiri and Chaudhuri [67] estimate defocus blur at edge locations, by modelling the defocus blur as a heat diffusion process. In contrast, Zhuo and Sim [109] model the defocus blur as a 2D Gaussian blur. The input image is re-blurred using a known Gaussian blur kernel and the gradient ratio between input and re-blurred images is calculated: This ratio gives the blur amount at edge locations; full depth map is then recovered using the matting interpolation. Good results are shown on different scenarios, although a user intervention is often adopted to solve ambiguities in depth estimation. Moreover, the authors do not consider to use depth maps to deblur the input image.

A re-synthesis application for defocused images is to synthetically increase the level of defocus, to reproduce the shallow depth of field found in large-aperture SLR photos. As Bae and Durand show, for the purpose of this simple application, defocus can be estimated sufficiently well just from cues in a single image [4].

Structured Light. Ma and Staunton describe in [58] a neural-network based approach to depth reconstruction, when a structured light is projected into the scene. The proposed solution uses two defocused image and is composed by two-stages: firstly objects are detected in 2D, and then 3D depth is estimated. The object detection is performed by a multi-resolution image segmentation to effectively isolate meaningful object regions from the background. Afterwards, a lower resolution image is fed into a three-layer artificial neural network as feature vectors and then processed to

give a depth estimate. Although the approach requires active illumination, the authors simulate it by gluing printed texture to the objects.

Another depth estimation method based on projecting structured light pattern into the scene is analyzed by Crofts in [18]. Edges profiles of the projected pattern are evaluated in order to obtain high-density depth maps. In the same work, the author proposes the concept of taking a succession of images whilst moving the light pattern, in order to increase the spatial resolution of depth estimates.

Active Illumination. Since DFD cannot estimate depth for textureless scene regions, some methods [32, 68, 66] use active illumination to project a texture onto the scene. In particular [32, 66] project a pattern on the scene, and its defocus is used to estimate the depth of the scene from a single image, albeit with blurred boundaries. The main goal of Girod and Adelson [32] is to determine whether the computed depth lies in front of, or behind, the focal plane (in other word, deciding the sign of eq.(3.7)). This is achieved by projecting a pattern consisting of asymmetric shapes. Then, to remove the pattern from the captured image, the authors suggest using low-pass filtering. However, such an approach will not work for textured scenes as it will significantly degrade the quality of the image. In contrast, [66] show that by projecting a grid of dots on the scene and using ratios of the acquired image with a set of calibration images, the dots can be removed even for textured scenes, without any noticeable loss of image quality.

The main limitation of techniques that use active illumination is that they can rarely be applied to outdoor scenarios.

A different approach to single image depth estimation is proposed by Saxena *et al.*[83, 84]. They present a probabilistic model to capture monocular cues and relation between different parts of the image. Besides defocus, monocular cues include texture variations (the texture of many objects look different at different distances), direction of edges (parallel lines appear to be tilted lines in an image) , light and shading.

2.2 Coded Aperture Systems

The concept of using a coded aperture was first introduced by Dicke [19] and Ables [1]. In the original formulation the single opening of a simple pinhole camera is replaced by many pinholes (called collectively *the aperture*) arranged randomly. The original motivation was to obtain an imaging system able to maintain the high angular resolution of a small single pinhole and, at the same time, to produce images that have a signal-to-noise ratio (SNR) commensurate with the total open area of the lens aperture. In the past, this technique has been employed notably in astronomy and medical imaging, especially for x-rays or gamma-rays, because traditional lenses could not be used at these wavelenghts [29]. Pinhole cameras, in fact, have a couple of advantages over lenses: they have infinite depth of field and they do not suffer from chromatic aberration. The biggest problem with pinhole cameras is that they let very little light through to the film or other detector. This problem can be overcome by making the holes larger, which unfortunately leads to a bigger blur and hence a decrease in resolution. The idea to solve this problem is to find a way to combine the rays entering the camera in a coded fashion that can be then separated by later decoding. This has been done with both optical mask and binary mask.

2.2.1 Optical Aperture Mask

Plenoptic cameras instantaneously capture the full light field entering the optical system: multiple view-points can so be collected in a single image and the user can adjust focus and aperture setting after the picture has been taken. The design implemented first by Ng [69], and successively in [31, 53, 8], trades spatial resolution to capture directional information about rays entering the optical system. This can also be seen as splitting the main lens aperture into a number of rectangular area and form a separate image from each of these sub-apertures. A typical drawback of this approach is a severely reduced spatial resolution, where the grid subdivision of the aperture results in a reduction that is quadratic in the number of samples along one axis. Also, the

system requires a more costly optical design (*e.g.*, a calibrated microlens array). An advantage of this approach is that the final image can be a simple linear combination of recorded data.

Another interesting way of splitting the light entering an optical system is to use beam splitters to replicate the optical path. Prisms and half-silvered mirrors are typical elements used to perform this task. In particular, McGuire *et al.* [63] use different aperture and focus settings to perform matting. There are two strongest limitations of these designs: 1) they usually require multiple sensors and 2) they lose light since they need to rely on occlusion by a mask to select a sub-region of the aperture. Another quite complex aperture decomposition by using optical elements is presented by Green *et al.*[35]: the aperture a split into a central disc and a set of concentric rings. The main problem for this optical system is that the results strongly depend on the accuracy of the calibration procedure.

An alternative and very common approach is called *wavefront coding*: The key idea is to use aspheric lenses to render the lens point spread function (PSF) depth-invariant. Then, shift-invariant deblurring with a fixed known blur kernel can be applied to sharpen the image [21, 44]. However, while the results are quite promising, the PSF is not fully depth-invariant and artifacts are still present in the reconstructed image.

2.2.2 Binary Aperture Mask

One of the first work using a binary mask is [39], where Hiura and Matsuyama propose two types of coded apertures and corresponding analysis algorithms: 1D Fourier analysis to acquire a depth map and a blur-free image from three defocused images taken with a two-hole aperture mask, and 2D convolution based model matching for the fast and precise depth measurement using a coded aperture with four holes. Experimental results show that the coded apertures improve the DFD range estimation capability for real world scenarios.

More recently, one of the most important contribution in this field comes from Levin *et al.*[50], who propose an algorithm to recover both depth and texture from a

single coded image. This is done in two steps: First, a deconvolution algorithm is applied to hypothesis planes and then, at each pixel, the plane that yields the smallest image reconstruction error is chosen. An important part of the work is that texture priors are explicitly formalized in a Bayesian framework and embedded in the deconvolution procedure. The estimated depth maps (sometimes corrected by the user) can be used for refocusing the input images, but their range and resolution is fairly limited. Another important work belongs to Veeraraghavan *et al.*[98], who designed a mask to be broadband in the Fourier domain, in order to improve out-of-focus deblurring. All these methods can handle a small amount of blur, but they fail when dealing with large scale of blur.

Examples of coded aperture systems include also the *off-axis camera* and the *programmable aperture camera*. Dou and Favaro [20] describes the former device as composed by a conventional camera where the aperture can be moved away from the centre of the lens. The latter camera, presented by Liang *et al.*[54] allows one to capture multiple images changing the shape of the lens aperture at each image. Both devices have been designed and used for reconstructing depth and texture of a static scene from multiple images, captured from the same view-point.

2.3 Summary

This chapter presents an overview of the most relevant methods that have been developed in the past to solve the problem of depth and all-in focus image estimation. This problem can be solved by using a conventional camera or a coded aperture device. However, coded aperture cameras have received much more attention in the last years, and recently it has been shown that their use improves performance of both depth estimation and image deblurring if compared with results from a circular aperture camera, when considering either a pair of images [107] or just a single image [106] as input.

This page has been left intentionally blank.

Chapter 3

Image Formation Model of a Coded Aperture Camera

Design is not just what it looks like and
feels like. Design is how it works.
Steve Jobs [1955-2011]

A coded aperture camera is a conventional camera with a mask on the lens. Therefore, we first need to understand how an image is formed in a conventional camera, before moving to study the coded aperture device.

The first part of this chapter analyzes the image formation model of a conventional camera. Every point of an object emits light rays, which travel through the camera lens to be finally projected into the pixels of the camera sensor (Section 3.1). On the sensor, the rays might be concentrated all in one pixel (the object is in-focus) or spread over several pixels (the object is out-of-focus): This effect is called *defocus* and it can be modelled in different ways, as described in Section 3.2.

The second part of the chapter examines what changes when a mask is placed on the main camera lens (Section 3.3). The study of the coded aperture model starts by considering a mask composed by a single off-axis hole, whose size has to be sufficiently large so that diffraction effect can be ignored (Section 3.3.1). The study pro-

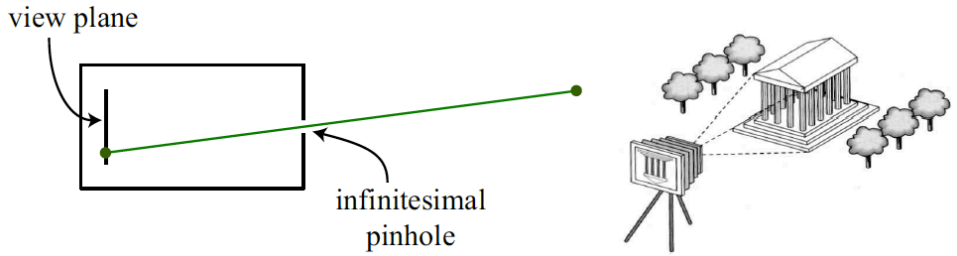


Figure 3.1: Pinhole camera model. Light rays from an object pass through a small hole to form an inverted (upside-down) image.

ceeds by adding in the model the contribution of several openings that compose the mask (Section 3.3.2).

Finally, the image formation model is used in Section 3.4 to develop a calibration toolbox that allows the user to find the optimal camera setting for a given scenario. The accuracy of the calibration procedure, and therefore of the derived model, is successfully tested on real data in Section 3.4.3.

3.1 Basic Analytic Models

3.1.1 Pinhole Camera

The simplest camera model available is the *pinhole* model, representing an ideal perspective camera where everything is in-focus. The pinhole model is specified by its centre of projection, which is coincident with the infinitesimal pinhole aperture (Figure 3.1). Every pixel on the view plane corresponds to a *single* ray from the scene: The entire captured image is therefore in-focus. Note that the pinhole does not redirect light from the scene, but simply restricts which rays reach the sensor. The most important setting for a pinhole camera is the distance from the pinhole to the sensor plane, which can be interpreted as a degenerate form of focus setting: For any possible distance, the pinhole camera will still generate an image perfectly in-focus, and moving the sensor plane has the side-effect of magnifying the image [37].

In practice, the pinhole model can be approximated by very small apertures (such

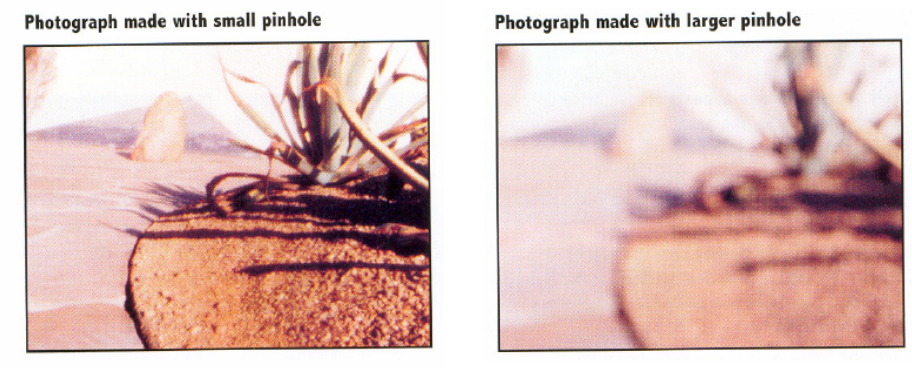


Figure 3.2: Pinhole camera images. Comparison between an image captured with a small pinhole (left) and an image captured with a large pinhole (right).

as f/22). However, diffraction limits the sharpness that can be achieved with small apertures (as it will be analysed in Section 3.3.1). Another limitation of small apertures is that they gather less light, meaning that they require long exposure times or strong external lighting. We can increase the amount of incoming light by using a larger pinhole, but the sharpness of the image will be reduced, as shown in Figure 3.2. Therefore, we need to introduce a lens in the model in order to refocus the image.

3.1.2 Thin Lens Model

Most modern cameras use a lens to focus light onto the view plane (i.e., the camera sensor). The use of lenses allows one to capture enough light in a period of time that is sufficiently short so that 1) the objects in the scene do not move and 2) the image is bright enough to show significant detail over a wide range of intensities and contrasts.

Lens models can be quite complex, especially for compound lens which are present in most cameras. In this section we consider the simplest case, widely known as the *thin lens model* [9, 97]. In the thin lens model, rays of light emitted from a point $\mathbf{P} = [x_1, y_1, z_1]$ in the scene, not too far from the optical axis[1], travel along paths through the lens, converging at a point $\mathbf{p} = [x_0, y_0, z_0]$ behind the lens. The key

[1]Specifically, the thin lens model requires that $\frac{x_1}{z_1}$ and $\frac{y_1}{z_1}$ are sufficiently small. Wide angle lenses cause problems for the model, but typical lenses used in digital cameras and considered in this thesis are fine, e.g., *50 mm* or more.

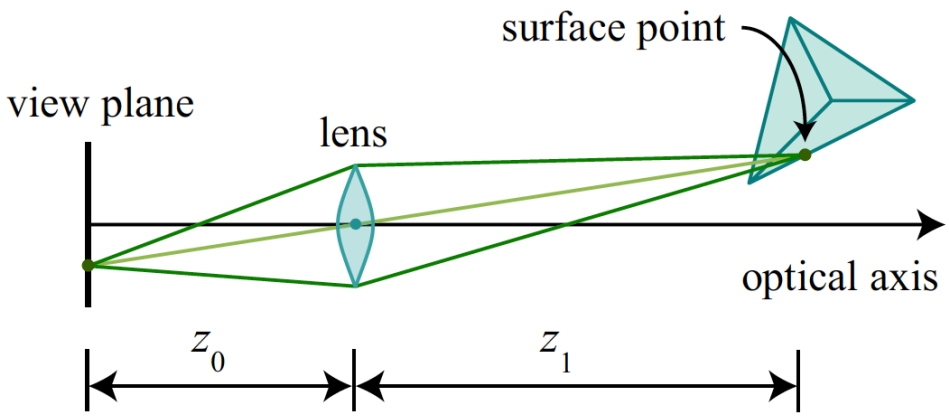


Figure 3.3: Thin lens model. Imaging of an object by the thin lens model.

parameter controlling this behaviour is called the *focal length* of the lens. The focal length F can be defined as the distance behind the lens to which light rays parallel to the optical axis, *i.e.*emitted from an infinitely distant source, will converge. More generally, if z_1 is the distance from the centre of the lens to a surface point **P** on an object, then, for a focal length F, the rays from **P** will be in focus at a distance z_0 behind the lens centre, where z_1 and z_0 satisfy the thin lens equation:

$$\frac{1}{F} = \frac{1}{z_1} + \frac{1}{z_0}. \tag{3.1}$$

Note that the rays going through the centre of the lens, also known as *principal rays*, are not deflected.

3.1.3 Relationship between Depth and Defocus

If the rays incident on the lens from a given 3D scene point do not converge to a unique point on the sensor plane, the scene point is considered to be *defocused*, and the extent of its defocus can be measured according to the footprint of these rays on the sensor plane. Conversely, a point on the sensor plane is defocused if not all rays that converge to that point originate from a single 3D point lying on the scene surface.

As initially defined in [22, 23, 72], the extent of an object's defocus can be related to

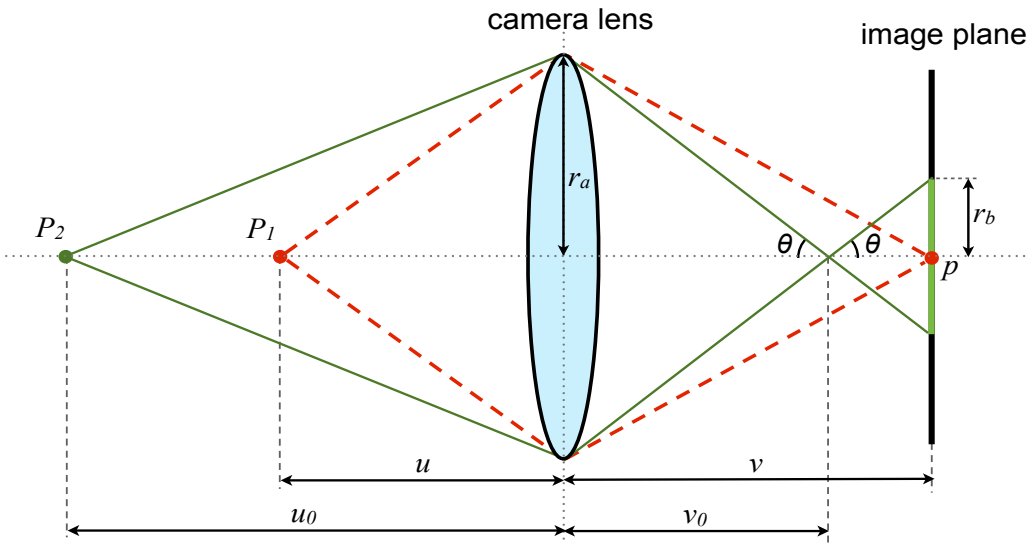


Figure 3.4: Depth/defocus relationship. The same object point, placed at different distances, will be recorded in the image sensor with different sizes of blur, depending on its distance from the focal plane.

its depth in an imaging system. Figure 3.4 illustrates the geometry of this relationship. The light rays from an object point P_1, placed at a distance u, pass through a spherical thin lens of radius r_a . As the rays from this point converge exactly on the pixel p of the image plane at distance v from the lens, then we say that the object point P_1 is *in focus* [18]. If we move the object point from P_1 to P_2, where the distance from the lens is $u_0 > u$, the rays converge at a point that is at distance $v_0 < v$ from the lens on the image side: Therefore, when the light rays from object point P_2 intercept the image plane (placed at distance v), the light energy is dispersed to form a *defocused blur*[1] of radius r_b.

A similar situation happens when the object point P_2 is moved to a distance $u_0 < u$: In this case the light rays converge at a distance $v_0 > v$, but they still generate a circular blur on the image sensor, representing the defocused image of the point P_2.

Given the focal length F of the lens, the parameters relative to the object point P_2

[1]The defocus blur will be fully described in Section3.2

(in Figure 3.4) can be used in the thin lens law, equation (3.1):

$$\frac{1}{F} = \frac{1}{u_0} + \frac{1}{v_0}. \tag{3.2}$$

This can be rearranged in an expression that provides the desired object distance u:

$$u_0 = \frac{Fv_0}{v_0 - F}. \tag{3.3}$$

For an object point where $u_0 > u$ (*e.g.*, point P_2 in Figure 3.4) and a blur circle is formed on the image plane, it can be stated that

$$\tan\theta = \frac{r_a}{v_0} = \frac{r_b}{v - v_0}. \tag{3.4}$$

Renaming u_0 as depth d [18], and combining equation (3.4) and equation (3.3), we obtain

$$d = \frac{Fr_a v}{r_a v - F(r_a + r_b)}. \tag{3.5}$$

Therefore, as described in equation (3.5), if we know the camera parameters, we can estimate the depth by measuring the size of the blur r_b.

In the same way, we can rewrite equation (3.5) in order to obtain the blur size of an object from its distance from the camera d:

$$r_b = r_a \left(\frac{vd - Fv - Fd}{Fd} - 1 \right). \tag{3.6}$$

For an object point placed at distance $u_0 < u$, we have $v_0 > v$, which introduces a sign "–" in the right term of equation (3.4). Hence, we can rewrite equation (3.6) in a more general form as

$$r_b = \pm\, r_a \left(\frac{vd - Fv - Fd}{Fd} - 1 \right), \tag{3.7}$$

where the "+" sign holds for object placed further than the focal plane u, while the sign "–" holds for objects placed closer to the lens ($u_0 < u$) [28].

3.2 Defocus Models

In the previous section we have studied how an object point is projected into the image sensor of the camera. There are two important cases: 1) if the object is placed at the focal plane, it is represented as a single point in the sensor (in-focus); 2) if the object is away from the focal plane, we have a defocused representation of it (out-of-focus).

The following sections recall the most common models used for representing the defocus blur. It starts with the blur from a single point, to then combine together the contribution from all points of the scene, obtaining the entire image that is formed in a conventional camera.

3.2.1 Analytic Models of the Blur Kernel

The blur kernel, or point spread function (PSF), describes how the light rays from an object point are dispersed once they reach the camera sensor; it can be seen also as the image brightness distribution produced by a point light source, placed at given distance d from the camera. Since the blur size changes with the location of the object point (as seen in Section 3.1.3), the PSF is denoted with the symbol h_d, to show this dependency. It is of common use to assume that the blur kernel h_d is normalized,

$$\int \int h_d(x, y) dx dy = 1, \tag{3.8}$$

i.e., all the light rays emitted from a given object point are contained in h_d.

On the assumption that a typical camera system has a circular aperture, the blurred image of the point light source is circular in shape. The two most commonly used analytic models for this type of blur are the pillbox function and the Gaussian function, which are illustrated in Figure 3.5 [17, 28].

Pillbox defocus model. Based solely on geometric optics, light intensity distribution within the blur circle is approximately constant. This model is generally known as the *pillbox function*. Under the idealization that the aperture is circular, the footprint of

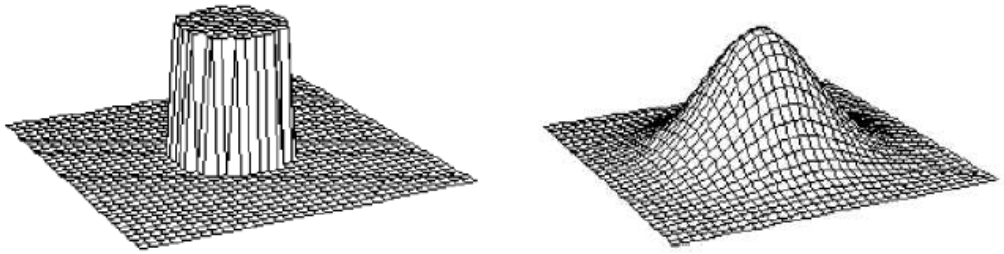


Figure 3.5: Structure of the Point Spread Function. Pillbox (left) and Gaussian (right) model used to represent the blur kernel.

a point on the image sensor will be also circular, leading to a cylindrical, or pillbox, model of defocus [85, 100]:

$$h_d(x,y) = \begin{cases} \frac{1}{\pi r_b^2} & \text{if } x^2 + y^2 \leq r_b^2 \\ 0 & \text{otherwise} \end{cases} \tag{3.9}$$

where r_b is the radius of the blur circle (see Figure 3.4).

Gaussian defocus model. Although first-order geometric optics predict that defocus within the blur circle should be constant, as in the pillbox function, the combined effects of such phenomena as diffraction, lens imperfections, and aberrations mean that a 2D circular Gaussian may be a more accurate model for defocus in practice [72, 27]:

$$h_d(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{3.10}$$

where σ is the standard deviation of the Gaussian.

3.2.2 Defocus as Linear Filtering

In computer vision, the dominant approach to defocus is to model it as a form of linear filtering acting on an ideal in-focus version of the image [37]. The advantage of using this model is the possibility of describing an observed defocused image, $\boldsymbol{g}$, as a simple convolution,

$$\boldsymbol{g} = \boldsymbol{h_d} * \boldsymbol{f}, \tag{3.11}$$

where $\boldsymbol{h_d}$ is the 2D blur kernel, and $\boldsymbol{f}$ is the ideal pinhole image of the scene. In these terms, the assumption in equation (3.8) indicate that the intensity of each pixel in $\boldsymbol{f}$ is still all present in the blurred image $\boldsymbol{g}$. The model of defocus as linear filtering follows from Fourier analysis applied to a fronto-parallel scene [88]. The blur kernel acts as a low-pass filter, so that as the image is defocused, contrast is lost and high frequencies are rapidly attenuated.

3.2.3 Spatially Variant Filtering

To relax the assumption that the scene consists of a fronto-parallel plane, we can model the blur kernel as spatially varying, i.e., $h_{d(x,y)}$, corresponding to a scene that is only locally fronto-parallel [6, 78, 79, 17, 28]. This results in a more general linear filtering,

$$g(x,y) = \int\int_{s,t} h_{d(s,t)}(x-s, y-t) f(s,t)\, ds\, dt\ , \tag{3.12}$$

where (s,t) is the 2D location of a pixel on the in-focus image f, while (x,y) represents a pixel of the burred image g. Equation (3.12) can be thought of as independently defocusing every pixel in the sharp image, f, according to varying levels of blur (and therefore varying levels of depth), and then integrating the results. Note that although this defocusing model is no longer a simple convolution, it is still linear, since every pixel $g(x,y)$ is a linear function of f [37].

In practice, smoothness priors are often introduced on the spatially variant blur $d(x,y)$, corresponding to smoothness priors on the scene geometry [78, 79]. These priors help to regularize the recovery of $f(x,y)$ from the image formation model of equation (3.12), and balance reconstruction fidelity against discontinuities in depth.

3.3 Coded Aperture Model

After having studied how an image is formed in the conventional camera, we now examine the variations on the image formation model when a mask is placed on the camera lens. Suppose the mask is composed by a single off-axis opening. In this

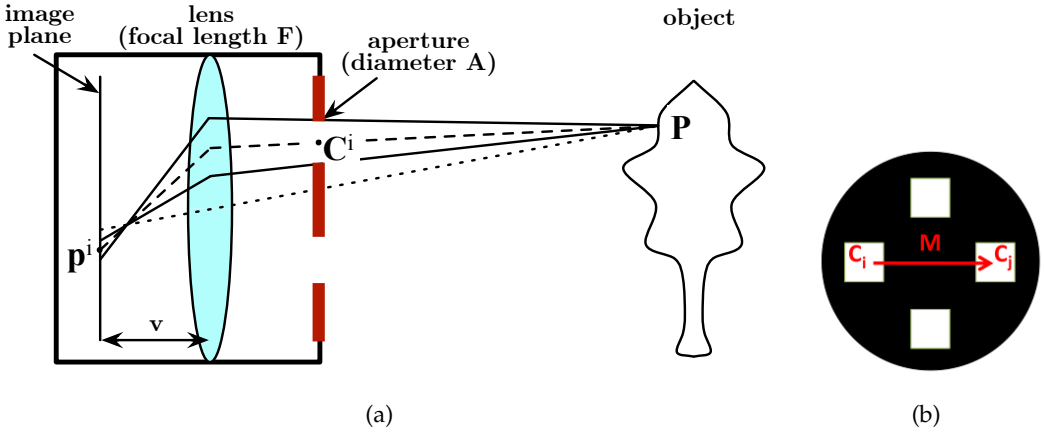


Figure 3.6: Geometry of coded aperture camera and example of mask. The model sketched in (a) is a 2D section of the camera; the red thick line represents the mask. An example of such a mask is shown in (b).

case, the analysis is similar to the off-axis camera model, introduced in [20]. Figure 3.6(a) sketches the coded aperture camera as a device composed of: 1) an image plane (camera sensor), 2) a thin lens of focal length F (camera lens), and 3) a mask formed by N apertures, where each aperture has diameter A and is centred in $\boldsymbol{C}^i = [C_x^i \; C_y^i \; C_z^i]^T \in \mathbb{R}^3$, $i = 1 \ldots N$. The distance image plane to lens is denoted by v.

Let $\mathbf{P} = [P_x \; P_y \; d]^T \in \mathbb{R}^3$ be a point in space lying on the object of interest; then the projection of $\mathbf{P}$ in the image plane through the aperture i is defined by the pixel $\mathbf{p}^i = [p_x^i \; p_y^i]^T$ as

$$\begin{bmatrix} p_x^i \\ p_y^i \end{bmatrix} = -\frac{v}{d} \begin{bmatrix} P_x \\ P_y \end{bmatrix} + \left(1 - \frac{v}{v_0}\right) \left(\begin{bmatrix} C_x^i \\ C_y^i \end{bmatrix} d - C_z^i \begin{bmatrix} P_x \\ P_y \end{bmatrix} \right) \frac{1}{d - C_z^i} \,, \tag{3.13}$$

where

$$v_0 = \frac{Fd}{d - F} \tag{3.14}$$

indicates the distance between the camera lens and the plane where the object $\mathbf{P}$ is in focus (see also the defocus model in Figure 3.4).

Notice that when the image is brought into focus, *i.e.*, when the image plane is at a distance $v = v_0$, the projection $\mathbf{p}_i$ coincides exactly with the prospective projection of

P in a pinhole camera (Section 3.1.1):

$$\begin{bmatrix} p_x^i \\ p_y^i \end{bmatrix} = -\frac{v}{d} \begin{bmatrix} P_x \\ P_y \end{bmatrix}. \tag{3.15}$$

Instead, when the image is out-of-focus, *i.e.*$v \neq v_0$, the point **P** generates a blur disc of diameter B, that can be computed as the distance of the projection of **P** through two opposite points on the aperture edge:

$$B = A \left| 1 - \frac{v}{v_0} \right| \frac{d}{d - C_z^i}, \tag{3.16}$$

which is identical to the well-known formula used in shape from defocus when $C_z = 0$ (see equation (3.7)), *i.e.*, when the aperture lies on the lens [17, 28].

3.3.1 Diffraction Effects

This analysis considers masks with openings sufficiently large so that diffraction can be ignored. Indeed, a plane wave of unit intensity and wavelength λ traveling through a circular aperture of radius r_a generates in the far field a Fraunhofer diffraction pattern, also called Airy disk [9], which can be written in terms of the ratio $\kappa = r_a/\lambda$ and the Bessel function J_1 of the first kind and of the first order, as

$$I(\theta) = \left(\frac{2J_1(2\pi\kappa \sin(\theta))}{2\pi\kappa \sin(\theta)} \right)^2 \tag{3.17}$$

where θ is the angle between the optical axis passing through the center of the circular aperture and the line between the center of the circular aperture and the observation point. Hence, in the coded aperture camera (which is diffraction-limited) one can consider the first zero of the Airy disk to define the radius of the diffracted beam r_β. This yields

$$r_\beta = 1.22\lambda F_\#, \tag{3.18}$$

where

$$F_{\#} = F/(2r_a) \tag{3.19}$$

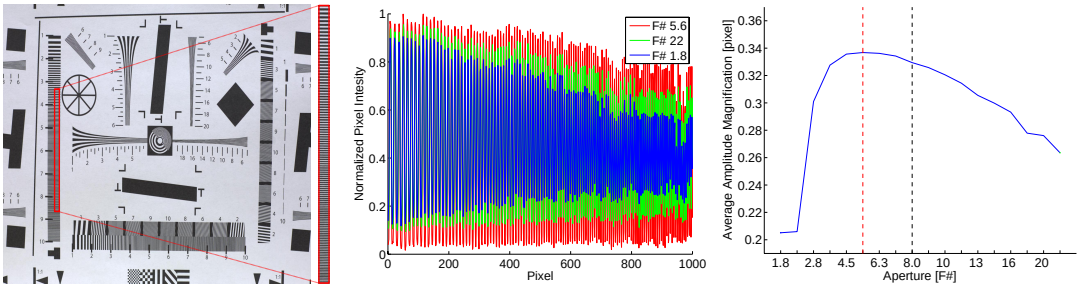
is the F-number of the camera, and F is the focal length of the main lens. By using the Rayleigh criterion, two point sources are considered distinct if they are separated by at least the radius of the Airy disk r_β. The size of a pixel on the sensor is denoted by γ. Then, for diffraction to be negligible, one needs

$$r_\beta \leq \gamma \, . \tag{3.20}$$

By rearranging the terms on both sides of the inequality (3.20), and substituting equation (3.18), the radius of the smallest opening in the mask is given by the following lower bound

$$r_a \geq 1.22\frac{F\lambda}{2\gamma} \approx 2.79mm \, , \tag{3.21}$$

where it is considered $F = 50mm$, $\lambda = 750nm$ (red visible light) and $\gamma = 8.2\mu m$. In other words, at any opening in the mask there must be a disk of radius $2.79mm$ or larger in order to avoid diffraction effects. Figure 3.7(a) displays a resolution chart captured with a conventional camera. A portion of the image is magnified at the right. The selected region contains a chirp signal that has low frequencies at the top and high frequencies at the bottom. Figure 3.7(b) shows the chirp signal captured with three different F# as a 1D plot where the frequency increases going from the left to the right. The plot shows clearly that when the aperture becomes too small, the captured image tapers more and more high frequencies due to diffraction effects. The same effect can be observed for the opposite change in the aperture: As one can see in Figure 3.7(c), as the aperture becomes too large, the captured image begins to taper high frequencies again. The optimal value for the F# predicted by Rayleigh criterion is shown with a black dashed vertical line, while the value found experimentally is shown as a red dashed vertical line.



(a) (b) (c)

Figure 3.7: Diffraction effects due to pupil intensity transmission changes. (a) Resolution chart (left) and magnification of the chirp signal (right) used to display the diffraction effect. (b) Chirp signal displayed as a 1D plot when frequencies increase going from left to right. The chirp signal is captured with three different F# so that one can appreciate the tapering effect at high frequencies due to a reduction of the aperture size. (c) The average amplitude magnification of the chirp signal in images captured with different aperture sizes (recall equation (3.19)). Notice the tapering effect when the aperture is either too big or too small. The black and the red dashed vertical lines in (c) indicate the values found theoretically (by using equation (3.21)) and experimentally respectively.

3.3.2 Superposition in Coded-Aperture Imaging

Under the assumption that the aperture mask is designed to make diffraction effects negligible, as described in the previous section, the model can now be extended to approximate an image generated when using with a generic mask.

Assuming that the scene is composed by locally fronto-parallel planes, as seen in Section 3.2.3, the image $\boldsymbol{g}$, generated by a single-hole aperture mask can be written as

$$g(\mathbf{p}) = \int h_d(\mathbf{p}, \mathbf{q}) f(\mathbf{q}) d\mathbf{q} \,, \qquad \forall \mathbf{p} \in \Omega \subset Z^2 \tag{3.22}$$

which is the vectorized form of equation (3.12), where $\mathbf{p} = [x \; y]^T$ is the pixel of the coded image $\boldsymbol{g}$ and $\mathbf{q} = [s \; t]^T$ is the pixel of the sharp scene $\boldsymbol{f}$, placed at a distance $\boldsymbol{d}$ from the camera. The pictures in Figure 3.8(a) and Figure 3.8(b) are two examples of images $\boldsymbol{g}$ obtained with such small apertures.

Suppose now that the aperture mask is composed of N identical openings. Because of the additive properties of light, the observed image can be written as the following

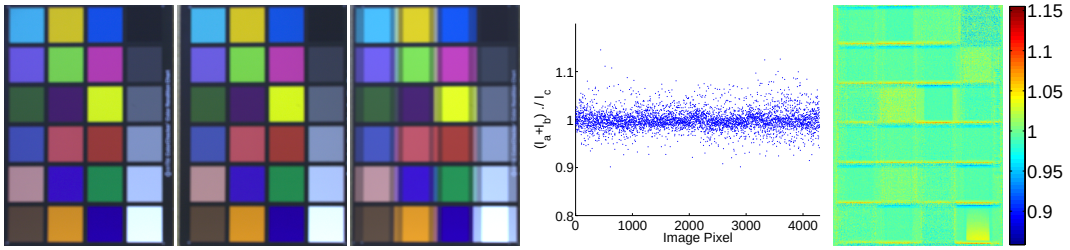


Figure 3.8: Superposition in coded-aperture imaging (linearity). Left: Images captured with an aperture mask composed by two horizontal holes: (a) image I_a obtained by keeping only the right hole open, (b) image I_b obtained with only the left hole open, (c) image I_c captured with both openings. **Right**: The ratio between the sum $I_a + I_b$ and the image I_c is shown in both graphs (d) and (e): in (d) the image has been reshaped as a row vector, while in (e) the pixel-wise ratio is shown as an image in pseudo-color. As one can see, the image obtained in (c) is very well approximated by the synthetic sum $I_a + I_b$. The main discrepancy between these images is due to noise, which is higher in dark regions.

linear combination [55]:

$$g(\mathbf{p}) = \int \sum_{n=1}^{N} h_d(\mathbf{p} + d(\mathbf{q})\Delta_n, \mathbf{q}) f(\mathbf{q}) d\mathbf{q} \,, \qquad \forall \mathbf{p} \in \Omega \tag{3.23}$$

where $\{\Delta_n\}_{n=1,\dots,N}$ denote the 2D centers of the N holes composing the mask. More in general, the model can be written in a more compact and realistic way as

$$g(\mathbf{p}) = \int h_d(\mathbf{p}, \mathbf{q}) f(\mathbf{q}) d\mathbf{q} + w(\mathbf{p}), \qquad \forall \mathbf{p} \in \Omega \tag{3.24}$$

by collecting all the effects of the mask in a single PSF, $h_d(\mathbf{p}, \mathbf{q})$, and by introducing zero-mean uncorrelated additive Gaussian noise $w(\mathbf{y}) \sim \mathcal{N}(0, \sigma^2)$.

The discrete form of equation (3.24) is

$$g(\mathbf{p}) = \sum_{q} [h_d(\mathbf{p}, \mathbf{q}) f(\mathbf{q})] + w(\mathbf{p}) \,, \tag{3.25}$$

which can be rewritten also with the matrix-vector notation

$$\boldsymbol{g} = \underbrace{\begin{bmatrix} \boldsymbol{h}_1 & \boldsymbol{h}_2 & \dots & \boldsymbol{h}_M \end{bmatrix}}_{\boldsymbol{H}_d} \cdot \left.\begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_M \end{bmatrix}\right\} \boldsymbol{f} + \boldsymbol{w} \,, \tag{3.26}$$

where M is the total number of pixels in the image and $\boldsymbol{h}_i$ is the PSF (ordered as a column vector) of the i-th pixel of the sharp image $\boldsymbol{f}$. The matrix $\boldsymbol{H_d}$ is sparse and has a block-Toeplitz structure [7]. Notice that, by ordering the images $\boldsymbol{g}$ and $\boldsymbol{f}$ as column vectors, the model can be expressed as a product of matrices, which is very fast to compute:

$$\boldsymbol{g} = \boldsymbol{H_d f} + \boldsymbol{w}. \tag{3.27}$$

Figure 3.8 shows that experimentally the above model is a reasonable approximation of the image formation process. Examples of PSFs that are typically used to approximate the image of a circular aperture are the Pillbox function and the Gaussian function, as seen in Section 3.2.1. The algorithms proposed in this thesis are not restricted to any such function.

3.4 Calibration of the Camera Parameters

The calibration procedure uses the coded camera model, introduced in Section 3.3, to find the camera setting that yields the best performance in both depth estimation and image deblurring. In Section 3.4.1 the camera parameters are introduced and linked to the image formation model, while Section 3.4.2 presents the Matlab-based calibration toolbox that has been developed. Finally, in Section 3.4.3 some experiments are carried out to show the accuracy of the calibration, and therefore of the coded aperture model described in this chapter.

3.4.1 Camera Parameters

Let **P** be a generic object point of the scene. Its projection through one of the N openings of the aperture mask generates a blur disc B. Such blur can be computed by using equation (3.16). Consider now the projections through all the N openings of the mask: The image generated by **P** is a combination of blurred discs that resemble the shape of the mask. This image corresponds to the PSF, as seen in Section 3.3.

Similarly to the blur size, the size of the PSF, which is denoted as S_{PSF}, can be defined (in pixel unit) by calculating the distance of the projections of **P** through the two furthest apertures in the mask, placed at a distance M (see Figure 3.6(b)):

$$S_{PSF} = \frac{1}{\gamma} M \left| 1 - \frac{v}{v_0} \right| \frac{d}{d - C_z^i}, \tag{3.28}$$

where γ is the physical size of a pixel in the camera sensor. Notice that the size S_{PSF} changes with d, as anticipated in Section 3.1.3. To determine if an object point $\mathbf{P}_1$ is closer or further that another object point $\mathbf{P}_2$ in the scene, it is enough to compare the size of their respective PSFs. Hence, the accuracy of a depth estimation method depends on the ability to discriminate PSFs. The purpose of the calibration procedure is to find the camera setting that maximises the PSF difference for a given scenario.

For the benefit of the reader, in the following there is a list of the parameters used in the calibration method, and their link with the coded camera model (see also Figure 3.6):

- **Depth range [min-max] -** d: minimum and maximum distance of the objects of interest from the camera lens.
- **Focal length -** F: focal length of the main lens;
- **Mask size -** M: measurement between the centres of the two furthest apertures in the mask.
- **Aperture size -** A : dimension of each hole that composes the aperture mask;

- **Pixel size -** γ: physical dimension of each pixel in the image sensor;
- **Distance lens-mask -** C_z: distance between the lens and where the mask is placed (the distance is negative if the mask is placed between the lens and the camera sensor).
- **Downsampled image -** λ: ratio between the dimension of the original image and the input image;
- **Subpixel -** α: minimum PSF scale difference (in pixels) that can be distinguished by the depth estimation algorithm (default is 1 pixel);
- **Number of depth levels**: number of depth levels in the captured scene that can be distinguished by the proposed algorithm in the ideal case. This amount is obtained by computing the difference between the biggest and the smallest PSF sizes generated by object points in the scene:

$$\#\, levels = \frac{S_{PSF}^{max} - S_{PSF}^{min}}{\alpha\lambda} + 1. \tag{3.29}$$

3.4.2 Matlab Calibration Toolbox

A Matlab-based calibration toolbox has been developed to obtain the best camera setting for a given scenario and the ideal depth resolution for such setting. The inputs of the procedure are the system parameters and the depth range, i.e., where the objects of interests are. In Figure 3.9 a screen shot of the calibration toolbox is shown. The user inserts the system parameters in the top part of the window, while the bottom part contains the output graphs. These graphs show how accuracy of the 3D estimation and the quality of the deblurring are affected by the camera setting.

The first graph at the left uses equation (3.29) to shows the number of depth levels that are possible to distinguish, depending on the position of the focal plane (the distance is measured starting from the camera lens). The graph at the centre is based on equation (3.16) and illustrates how the size of the blur of each hole in the mask

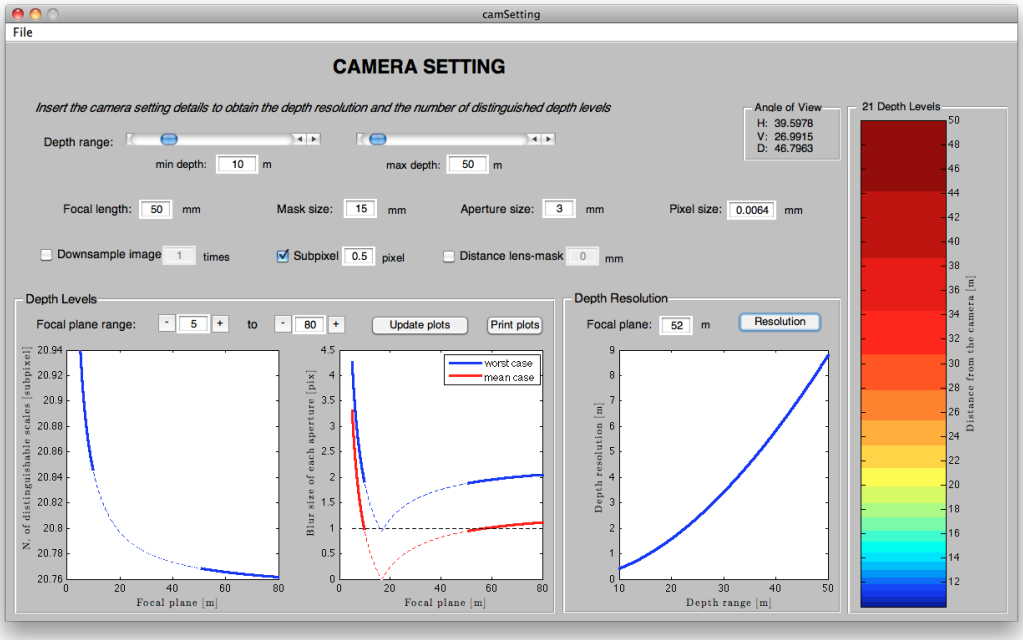


Figure 3.9: Screen-shot of the calibration toolbox.

changes with the focal plane position: The red line has been calculated by averaging the blur sizes generated by objects uniformly distributed along the z-axis. The blue line, instead, considers objects are at the furthest possible location from the camera. The dashed black line indicates when the blur size corresponds to 1 pixel. The quality of the image is preserved more when the blur of each single hole stays close to the dashed black line. In both graphs, the dashed lines indicate the interval where objects of interest are. Notice that placing the focal plane within this range would result in ambiguous 3D reconstructions, as outputted in equation (3.7). Hence, the focal plane will be set before or after the range of interest when capturing the datasets used in this thesis.

The bottom-right part of the window illustrates the *depth resolution*: For each depth $\tilde{d}$ the graph plots (in metres) the next depth increment that results in a detectable change of PSF size[1]. In formulas, the resolution of a depth $\tilde{d} \in [D_{min}, D_{max}]$ is given

[1]A change in PSF size is detectable if it is greater then $\alpha\lambda$ pixels.

by:

$$R(\tilde{d}) = \left| \left\{ d \mid d \geq \tilde{d},\ \left| S^{d}_{PSF} - S^{\tilde{d}}_{PSF} \right| < \alpha\lambda \right\} \right|. \tag{3.30}$$

The same measurement is shown in a different format at the far right, where the vertical colored stripe simulates the depth range (the minimum distance from the camera is placed at the bottom): Objects placed within the same color band are not distinguishable by the depth estimation algorithm.

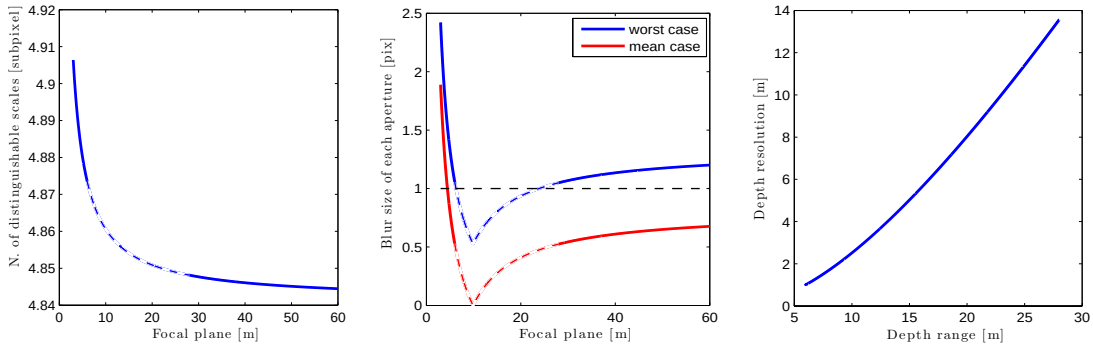
3.4.3 Experimental Validation

To show the accuracy of the coded aperture model previously described, the measurements obtained from the calibration toolbox have to be compared with real data. For a robust evaluation, the system is tested on different types of scenario, *i.e.*, different depth ranges and camera settings. All the images are captured in a controlled environment, where the distance from the camera to each object in the scene is known.

The coded aperture system used in this experiments is composed by a Canon EOS-5D Mark II camera body, a Canon 50mm f/1.4 lens, and the mask shown in Figure 3.6(b), which is placed on the main lens of the camera. In this case, the choice of the aperture mask is due to the fact that its shape makes the measurement of the PSF in the image easier. The size of the mask (M in the coded aperture model) is $11mm$ and is composed by square apertures each of $3mm$ diameter. The camera has a 36×24 mm CMOS sensor and, for this experiments, the images are captured with a resolution of 1920×1080 pixels, which makes the size of each pixel 18.72 μm. In these experiments the images are used at their original resolution ($\alpha = 1$) and sub-pixel accuracy is not considered ($\lambda = 1$).

Corridor Dataset [6m - 28m]

The first scene is a corridor and the depth range goes from $6m$ to $28m$. Figure 3.10 reports the three graphs given by the calibration toolbox as output to the parameters of the system previously described. As already described in details in section 3.4.2, in the first graphs (Figure 3.10(a)) the number of levels is computed by using equation (3.29).



(a) (b) (c)

Figure 3.10: Corridor dataset - graphs. (a) Number of possible depth levels (or difference in PSF size) depending on the position of the focal plane; (b) Blur size of each aperture in the mask; (c) Depth resolution when the focal plane is set at $3m$ from the camera (there is almost no difference when we set the focal plane to be further than $30m$.

	Corridor (a)		**Corridor (b)**		**Reindeer**	
	meas.	comp.	meas.	comp.	meas.	comp.
# levels	5.5	4.9	5.0	4.8	15.0	14.7

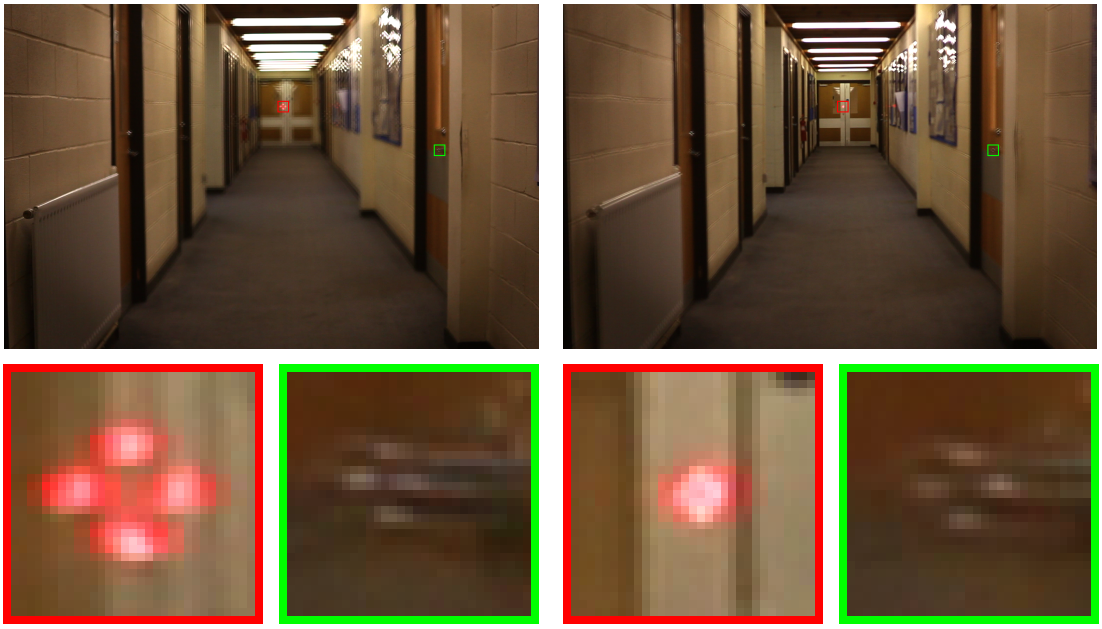
Table 3.1: Comparison between measured (meas.) and computed (comp.) number of depth levels for different datasets.

The number of possible depth levels corresponds to the difference, in pixels, between the PSF size at the closest object to the camera and the PSF size at the furthest point in the scene:

$$\#\ levels = S_{PSF}^{max} - S_{PSF}^{min} + 1. \tag{3.31}$$

In Figure 3.11 we show the difference on changing the position of the focal plane in the same scenario. In Figure 3.11(a) the focal plane is placed at $3m$ from the camera while in Figure 3.11(b) it is set to be after the scene ($35m$ about). The first case falls just at the left of the dashed lines in the graphs (a) and (b) of Figure 3.10, while the latter case corresponds to a point at the right-hand side of them.

The sizes of the PSF (S_{PSF}^{max} and S_{PSF}^{min}) have been manually measured at the closest and at the furthest object; then equation (3.31) is computed and the output is compared with the values given by the toolbox and reported in the graphs. This comparison is reported for different datasets in Table 3.1.

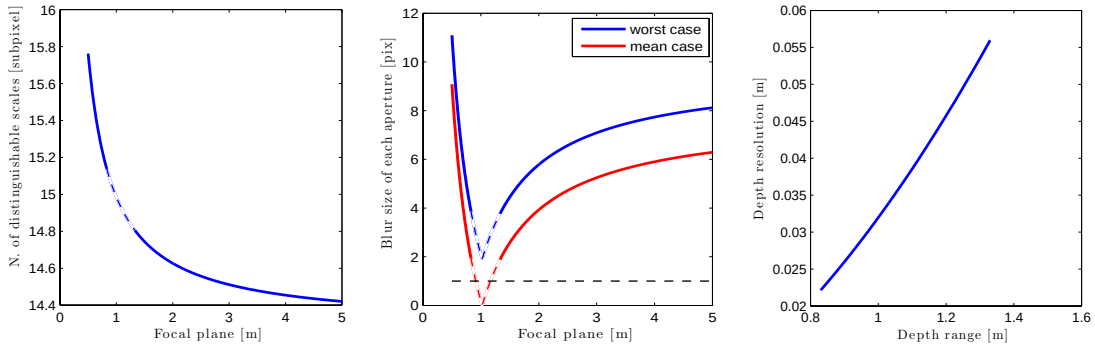


(a) (b)

Figure 3.11: Corridor dataset - real images. (a) Focal plane is set to be at 3m from the camera; (b) Focal plane is placed after the scene of interest. **Top row:** coded images of the scene; **Bottom row:** PSF from the furthest (red box) and from the closest (green box) object of interest in the scene.

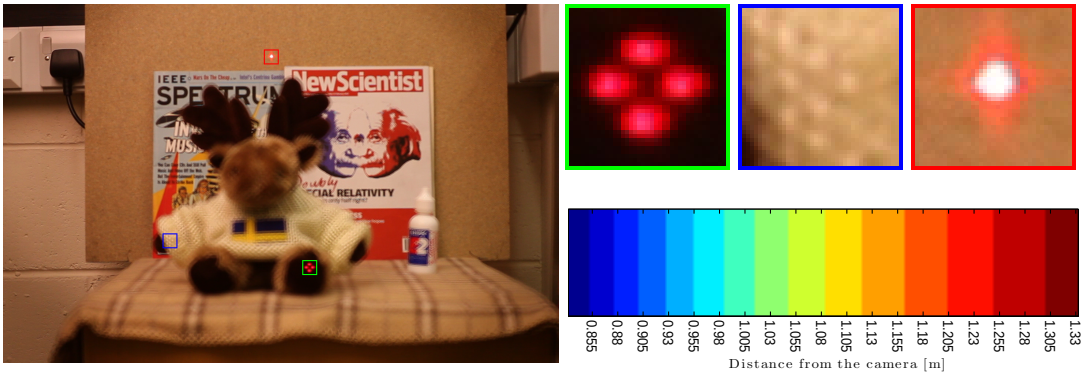
Reindeer Dataset [0.83m - 1.33m]

The second dataset is placed closer to the camera and the depths vary from 0.83m to 1.33m. The output graphs are shown in Figure 3.12. For this scenario only an image is taken (see Figure 3.13(a)), by placing the focal plane at a distance of 1.60m. The sizes of the PSF have been manually measured in order to compare the number of possible levels with the output of the toolbox (see Table 3.1). Figure 3.13(b) reports, at the top, three PSFs extracted from the captured image (the green and the red regions are respectively the closest and the furthest points of interest) and, at the bottom, the depth resolution chart. The blue-framed PSF is extracted from the arm of the teddybear, which is placed at 0.93m from the camera; the size of the PSF is 14 pixel, which is 4 pixels smaller than the first PSF (green box), placed at a depth of 0.83m. These measurements correspond exactly to the output of the depth resolution chart: an object at 0.93m differs of 4 levels from an object placed at 0.83m from the camera.



(a) (b) (c)

Figure 3.12: Reindeer dataset - graphs. (a) Number of possible depth levels (or difference in PSF size) depending on the position of the focal plane; (b) Blur size of each aperture in the mask; (c) Depth resolution when the focal plane is set at 1.60m from the camera.



(a) (b)

Figure 3.13: Reindeer dataset - images. (a) captured coded image with focal plane at 1.60m. (b) **Top row:** images of the PSF at three different depths (green-0.83m, blue-0.93m, red-1.33m); **Bottom row:** depth resolution chart (depths belonging to the same color band cannot be distinguish).

The same can be shown with the furthest PSF (red box), whose size is about 4 pixels.

3.5 Summary

In this chapter the image formation model of a coded aperture camera has been derived, starting from the model of a conventional camera. When a binary mask is placed on the lens, the shape of the blur generated by the camera represents the pattern of

the aperture mask at different scale, depending on the distance from the focal plane. One of the most important parameters is the size of each opening, which cannot be too small, otherwise diffraction would make the blur more difficult to identify.

The model has been implemented in a calibration toolbox, that allows one to simulate the number of depth levels that can be distinguished in a given scenario. The calibration procedure can then be used to map the identified blurs to real depth values (distance from the camera). Tests on real data have shown that the given image formation model is a reasonable approximation of what happens on a real device.

This page has been left intentionally blank.

Chapter 4

Depth and Image Estimation from a Single Coded Image

We cannot change the cards we are dealt,
just how we play the hand.
Randy Pausch [1960-2008]

This chapter introduces the key problem tackled by this work: Depth and all-in-focus image reconstruction from a single coded image. The problem is formulated in a Bayesian framework (Section 4.1), using the image formation model described in the previous chapter. Section 4.2 recalls the most important methods that have been previously used to solve this task, illustrating both their achievements and their limits. For the sake of clarity, the formulation of such methods is re-written by using the notation introduced in this thesis. The subsequent Section 4.3 introduces two novel approaches, which aim to solve the problem of 1) depth estimation alone and 2) both depth and image estimation; these approaches will be described in depth in Chapter 5 and Chapter 6 respectively.

Since the depth estimation (or blur identification) depends on the blur pattern, all the binary masks previously used in literature are described and illustrated in Section 4.4.

4.1 Problem Formulation in a Bayesian Framework

If one is given the depth map $\boldsymbol{d}$, the sharp image $\boldsymbol{f}$, and the camera parameters that define the aperture mask, $\boldsymbol{H_d}$, a realistic coded-aperture image can be simulated by computing equation (3.27), which is reported here

$$\boldsymbol{g} = \boldsymbol{H_d f} + \boldsymbol{w}.$$

In this manuscript, the goal is to address the inverse problem: Given a single coded-aperture image $\boldsymbol{g}$ and the camera parameters, one seeks for the depth map $\boldsymbol{d}^*$ and the sharp image $\boldsymbol{f}^*$ that generate $\boldsymbol{g}$. The main challenge in this inversion is ill-posedness [36]. Or rather, the number of unknowns largely outnumbers the number of measurements, and this results in multiple solutions that yield the same image $\boldsymbol{g}$. For instance, one can always consider the depth map as a plane in focus and the sharp image $\boldsymbol{f} \equiv \boldsymbol{g}$. Therefore, the solutions must be restricted to belong to a certain family of functions. This is somehow equivalent to introducing priors on what solutions one expects to obtain. Such priors are denoted via the probability distribution densities, which express an uncertainty about the depth, $p(\boldsymbol{d})$, or about the sharp image, $p(\boldsymbol{f})$. The Bayesian formulation of the posterior can be explicitly written as

$$p(\boldsymbol{d}, \boldsymbol{f}|\boldsymbol{g}) \propto p(\boldsymbol{g}|\boldsymbol{f}, \boldsymbol{d}) p\left(\boldsymbol{f}\right) p\left(\boldsymbol{d}\right) \ , \tag{4.1}$$

where equation (3.27) defines $p(\boldsymbol{g}|\boldsymbol{f}, \boldsymbol{d})$ as a Gaussian distribution with mean $\boldsymbol{H_d f}$ and the same covariance as that of $\boldsymbol{w}$

$$p\left(\boldsymbol{g} \,|\, \boldsymbol{f}, \boldsymbol{d}\right) = \mathcal{N}\left(\boldsymbol{g} \,\middle|\, \boldsymbol{H_d f}, \sigma^2 \boldsymbol{I}\right). \tag{4.2}$$

The inversion problem can then be posed as the following maximum a posteriori (MAP) problem

$$\boldsymbol{d}^*, \boldsymbol{f}^* = \underset{\boldsymbol{d},\boldsymbol{f}}{\operatorname{argmax}}\, p\left(\boldsymbol{d}, \boldsymbol{f} \mid \boldsymbol{g}\right) \tag{4.3}$$

$$= \underset{\boldsymbol{d},\boldsymbol{f}}{\operatorname{argmax}}\, p\left(\boldsymbol{g} \mid \boldsymbol{f}, \boldsymbol{d}\right) p\left(\boldsymbol{f}\right) p\left(\boldsymbol{d}\right). \tag{4.4}$$

When applying a log-likelihood to equation (4.4), the problem is expressed as a minimization of a functional

$$\boldsymbol{d}^*, \boldsymbol{f}^* = \underset{\boldsymbol{d},\boldsymbol{f}}{\operatorname{argmin}} \left[\underbrace{\sigma^{-2}|\boldsymbol{g} - \boldsymbol{H_d f}|^2}_{E_{data}(\boldsymbol{d},\boldsymbol{f})} + \underbrace{log(p(\boldsymbol{f})) + \log(p(\boldsymbol{d}))}_{E_{prior}(\boldsymbol{d},\boldsymbol{f})} \right], \tag{4.5}$$

which is composed by two terms, E_{data} and E_{prior}. The former term is also called *fidelity term* and it is based on the error between the measurements and the model assumed for representing the data (equation (4.2)). The latter term contains the priors on depth and texture.

Solving the task above requires estimating both $\boldsymbol{d}$ and $\boldsymbol{f}$. Before presenting the novel approaches of this thesis, the next section examines two important methods that have recently tackled the same problem in the field of coded aperture imaging.

4.2 Previous Approaches

The most important contributions to solve the problem of depth and texture estimation from a single coded image come from Veeraraghavan *et al.*[98] and Levin *et al.*[50]. The two methods use the same approach to tackle the problem described in equation (4.5). The approach can be divided in three parts: estimation of the hypothesis plane, depth reconstruction, and image estimation. In this section, we resume the main steps of these two methods, adapting their formulation to the notation that has been introduced in this thesis.

Suppose that the depth of the scene can be discretized in T levels, $d_1, \ldots, d_T$, and each depth level corresponds to a blur scale. Firstly, the authors consider a simplified version of equation (4.5) to deblur the given image using different blur kernels

$$\boldsymbol{f}_k^* = \underset{\boldsymbol{f}}{\operatorname{argmin}} \left[E_{data}(d_k, \boldsymbol{f}) + E_{prior}(\boldsymbol{f}) \right], \quad \text{with } k = 1, \ldots, T \,, \tag{4.6}$$

obtaining a sharp image $\boldsymbol{f}_k$ for each hypothesis-plane k. The term E_{prior} contains only texture prior, which is based on the concept that real-world images obey heavy-tail distributions in their gradients, as analyzed in [71]

$$p(\boldsymbol{f}) = e^{-\alpha |\Delta_x \boldsymbol{f}|^\gamma} \,, \tag{4.7}$$

with $\gamma = 1$ for [98], and with $\gamma = 0.8$ for [50].

In a blurred image the high spatial gradients are suppressed, therefore the tail of the gradient distribution is also suppressed. To overcome this problem, Veeraraghavan *et al.*[98] use the fourth-order moment (kurtosis) of gradients as a statistic for characterising the gradient distribution. Then the regularization term in [98] is defined as

$$E_{prior}(\boldsymbol{f}) = - \left[Kurt(\Delta_x \boldsymbol{f}) + Kurt(\Delta_y \boldsymbol{f}) \right] \tag{4.8}$$

Deblurring at an incorrect scale, larger than the correct scale, introduces high frequency deconvolution artifacts in $\boldsymbol{f}$. This may increase the gradient kurtosis, thereby decreasing E_{prior}.

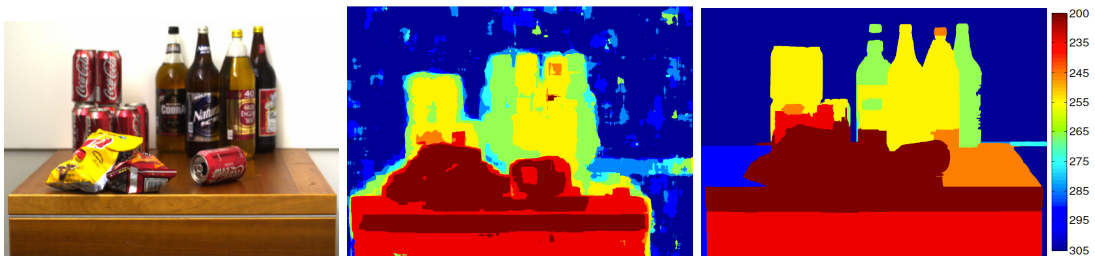
The depth is then estimated at each pixel $\mathbf{p}$ by the following minimization

$$\boldsymbol{d}^*(\mathbf{p}) = \underset{k}{\operatorname{argmin}} \, \lambda_k \left[|\boldsymbol{g}(\mathbf{p}) - \boldsymbol{H}_{d_k} \boldsymbol{f}_k^*(\mathbf{p})|^2 + E_{prior}(\boldsymbol{d}) \right] \,, \tag{4.9}$$

where the weights λ_k are constant in [98]; while in Levin *et al.*[50] they are learnt to minimize the scale classification error on a set of training images having a known depth profile. Both methods implement the minimization in equation (4.9) using an alpha-expansion graph-cut procedure [11]. Example of results on depth estimation

(a) Input coded image (b) Result without user correction (c) Result after user correction

Figure 4.1: Results presented by Veeraraghavan *et al.*[98]. (a) Input image captured with a coded aperture camera; (b) Depth map obtained without any user correction; (c) Final depth map after user correction.



(a) Input coded image (b) Result without user correction (c) Result after user correction

Figure 4.2: Results presented by Levin *et al.*[50]. (a) Input image captured with a coded aperture camera; (b) Depth map obtained without any user correction; (c) Final depth map after user correction.

obtained by these two methods are illustrated in Figure 4.1 and Figure 4.2.

Once the depth map is estimated, the sharp image $\boldsymbol{f}$ can be obtained by selecting the value of a pixel $\mathbf{p}$ from one of the hypothesis-planes $\boldsymbol{f}_k$, where the index k is given by the depth map $\boldsymbol{d}^*$.

4.2.1 Limitations

The main drawback of previous approaches is that they are limited to small amounts of blur. The benefit of such restriction is that errors in the depth estimate do not strongly affect the restored all-in-focus image. This however also limits the extension of the depth of field of a coded-aperture camera and the ability to exploit 3D information.

When dealing with a wide range of depth levels the hypothesis-plane method suffers from nonlocal artifacts introduced by structures that lie at depths different from the hypothesis being tested. As one can observe in Figure 4.3(a), this effect is not

(a) (b) (c)

Figure 4.3: Propagation of artifacts in the image reconstruction method. The images the reconstruction of the image for three hypothesis-planes: (a) Image restored with hypothesis plane in front of the object. (b) Image restored with hypothesis plane at the object. (c) Image reconstructed with hypothesis plane at the background.

very strong for small depth variations (*i.e.*, small amounts of blur). However, points on the background in Figure 4.3(c) are largely affected by artifacts introduced by nonlocal structures, although the correct hypothesis plane has been used. This encourages the algorithm to have a bias towards small amounts of blur.

Since the hypothesis-planes $\boldsymbol{f}_k$ are affected by this artifacts, the minimization of the error in equation (4.9) for depth estimation is not reliable, when one considers a wide range of depth values.

4.3 Novel Approaches

This section introduces two methods for restoring high quality depth maps over a wide range of depth levels. Starting from the formulation of the problem in equation (4.3), the two approaches make different assumptions about priors, but both of them obtain a formulation for the problem of depth estimation alone that avoids the image reconstruction.

In Chapter 5 we focus our study on a generic set of aperture masks composed by a small number of holes of identical size (e.g, the aperture masks in Figure 4.4(a) and Figure 4.4(b)). This allows one to consider only a few pixel in the computation,

instead that an entire patch. By reducing the estimation of the original sharp image to the local space-varying statistic of the texture, we obtain a novel method to directly estimate only depth, whilst still accounting for the statistics of the sharp image. This problem, denominated *shape from coded aperture*, can be written as

$$\boldsymbol{d}^* = \underset{\boldsymbol{d}}{\operatorname{argmax}}\, p\left(\boldsymbol{d} \,|\, \boldsymbol{g}, \boldsymbol{A}\right) \tag{4.10}$$

where $\boldsymbol{A}$ represents the statistics of the sharp texture $\boldsymbol{f}$.

In Chapter 6, instead, we have a more general algorithm, since we do not make any restriction on the type of coded aperture and we do not make any assumption on the texture statistics, but we learn it from natural images. We devise a novel method to reduce the maximization (4.3) as the following equivalent coupled problems

$$\begin{cases} \boldsymbol{d}^* &= \underset{\boldsymbol{d}}{\operatorname{argmax}}\, p\left(\boldsymbol{d} \,|\, \boldsymbol{g}\right) \\ \boldsymbol{f}^* &= \underset{\boldsymbol{f}}{\operatorname{argmax}}\, p\left(\boldsymbol{f} \,|\, \boldsymbol{g}, \boldsymbol{d}^*\right). \end{cases} \tag{4.11}$$

By solving (4.11) instead of (4.3) we can choose whether to recover depth alone or both depth and sharp texture, without trading off optimality for computational efficiency.

Examples of aperture masks recently designed for coded-aperture imaging and that we consider in the next Chapters are shown in Figure 4.4 and described in the next section.

4.4 Aperture Masks in Literature

One of the earliest pattern designs in astronomy is the Modified Uniformly Redundant Arrays (MURA) [34] for which a simple coding and decoding procedure was devised (see one such pattern in Figure 4.4(h)). The MURA consists of nearly 50% open space. Hiura and Matsuyama [39] use the two-hole (Figure 4.4(a)) and the 4-hole (Figure 4.4(b)) aperture masks for depth estimation from multiple coded images. Another interesting design, based on annular masks (Figure 4.4(c)), has also been pro-

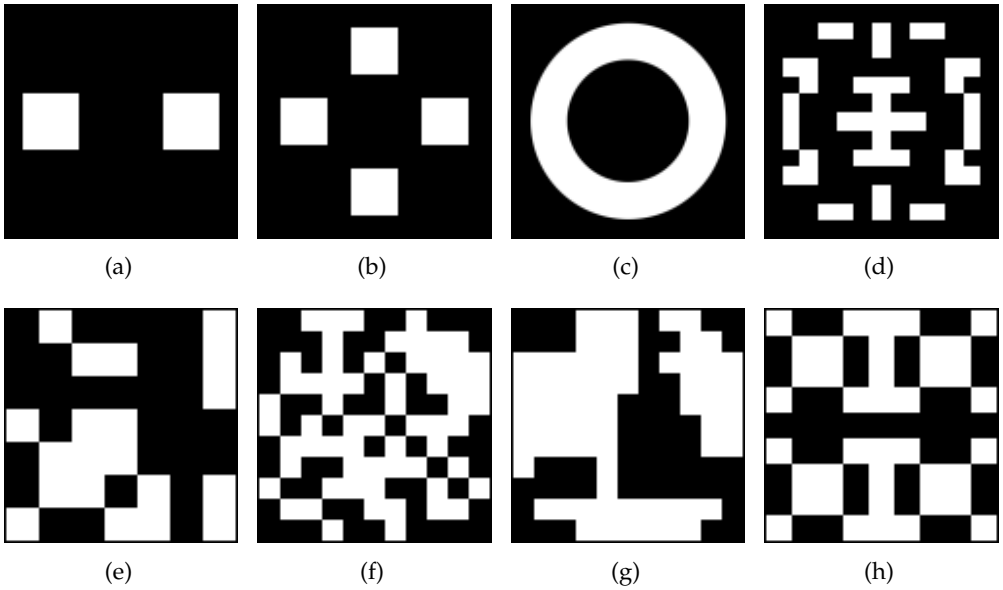


Figure 4.4: Coded aperture patterns. All the aperture patterns we consider in this work. (a) and (b) aperture masks used by Hiura and Matsuyama [39]; (c) annular mask used by McLean [64]; (d) pattern proposed by Levin *et al.*[50]; (e) pattern proposed by Veeraraghavan *et al.*[98]; (f) and (g) aperture masks used by Zhou *et al.*[106]; (h) MURA pattern used by Gottesman and Fenimore [34].

posed in [64], and successively exploited for the purpose of depth estimation by Farid [24]. Coded patterns have also been used to design lensless systems, but these systems require either long exposures or are sensitive to noise [110]. More recently, aperture coding has been used to preserve high spatial frequencies in blurred images so that deblurring is well-posed: for this goal Veeraraghavan *et al.*[98] propose the mask in Figure 4.4(e). A study on good apertures for image deblurring via Wiener filtering has instead led to novel designs [106]: in this thesis, two of their best performing aperture masks are considered (Figure 4.4(f) and Figure 4.4(g)). Although these masks are presented as optimal for image deblurring, they have a very poor performance in depth estimation, as one will see in Chapter 6.

Finally, image deblurring and depth estimation with a coded aperture camera has also been demonstrated by Levin et al. [50]; One of their main contributions is the design of an optimal mask (Figure 4.4(d)). This pattern has a very good performance

for depth estimation (see results in Chapter 6), especially for the methods based on deconvolution, like those described in Section 4.2.

4.5 Summary

This chapter presented, in a Bayesian framework, the problem of depth and all-in-focus image reconstruction from a single image. The most important previous approaches to this problem comes from Veeraraghavan *et al.*[98] and Levin *et al.*[50]. Both depth estimation methods are based on deconvolution, which creates artifacts on the hypothesis-planes when one deals with a wide range of depths. This limitation can be overcome if the image reconstruction is avoided for depth estimation purpose only, as shown in the novel methods that have been introduced here and will be described in detail in the next chapters. Examples of aperture masks, that have been found to be optimal for previous methods, are also illustrated here.

This page has been left intentionally blank.

Chapter 5

Shape from Coded Aperture for Simple Patterns

Fundamentals, fundamentals, fundamentals.
You've got to get the fundamentals down because
otherwise the fancy stuff isn't going to work.
Randy Pausch [1960-2008]

This chapter presents an analysis and a novel algorithm to estimate depth from a single image captured by a coded aperture camera. Unlike previous approaches, which need to recover both sharp image and depth, we consider directly estimating only depth, whilst still accounting for the statistics of the sharp image. The problem is formulated in a Bayesian framework, which enables a reduction of the estimation of the original sharp image to the local space-varying statistics of the texture. This yields an algorithm that can be solved via graph cuts (without user interaction). Performance and results on both synthetic and real data are reported and compared with previous methods.

5.1 Shape from Coded Aperture

The aperture masks considered in this chapter consist of a pattern that is composed of N squared openings, each offset by $\mathbf{\Delta_i}$, $i = 1 \dots N$. As studied in Section 3.3.2, the image $\boldsymbol{g}$ captured by a coded aperture camera with such a mask can be written as the linear combination of N views:

$$g(\mathbf{p}) = \frac{1}{N} \int \underbrace{\left(\sum_{i=1}^{N} \delta_d(\mathbf{p} + \boldsymbol{d}(\mathbf{p})\mathbf{\Delta_i}, \boldsymbol{q}) \right)}_{h_d(\mathbf{p},\boldsymbol{q})} \boldsymbol{f}(\boldsymbol{q}) \, \mathrm{d}\boldsymbol{q} + w(\mathbf{p}), \tag{5.1}$$

where $\mathbf{p}$ is a pixel of the image $\boldsymbol{g}$, $\boldsymbol{q}$ is a point of the object, and w is a zero-mean uncorrelated additive Gaussian noise $w(\mathbf{p}) \sim \mathcal{N}(0, \sigma^2)$.

5.1.1 Image Prior Model

Similarly to [16], an image prior is defined based on a set of P filtered versions of the original image $\boldsymbol{f}$:

$$\hat{\boldsymbol{f}}_k = \boldsymbol{C}_k \boldsymbol{f}, \qquad k = 1, \dots, P. \tag{5.2}$$

The operators $\boldsymbol{C}_k$ are zero mean conditional high-pass filters and each one of them is used to impose a particular constraint on the restored image $\boldsymbol{f}$.

Since $\boldsymbol{g}$ is Gaussian distributed (as defined in equation (4.2)) and $\boldsymbol{C}_k$ is a linear operator, the *commutative property*[1] can be utilised to obtain that $\hat{\boldsymbol{g}}_k = \boldsymbol{C}_k \boldsymbol{g} = \boldsymbol{H_d} \hat{\boldsymbol{f}}_k + \boldsymbol{C}_k \boldsymbol{w}$ is also a Gaussian distributed, and its conditional distribution is given by

$$p\left(\hat{\boldsymbol{g}}_k \,\middle|\, \hat{\boldsymbol{f}}_k, \boldsymbol{d}\right) = \mathcal{N}\left(\hat{\boldsymbol{g}}_k \,\middle|\, \boldsymbol{H_d} \hat{\boldsymbol{f}}_k, \boldsymbol{C}_k \sigma^2 \boldsymbol{I}\right). \tag{5.3}$$

The likelihood of our prior assumes that the k^{th} filtered versions of the sharp image $\boldsymbol{f}$

[1] Strictly this only holds for planar scenes; however we find this is a reasonable approximation if we work with locally frontal-planar patches.

follows a Gaussian distribution with zero mean

$$p\left(\hat{\boldsymbol{f}}_k \mid \boldsymbol{A}_k\right) = \mathcal{N}\left(\hat{\boldsymbol{f}}_k \,\middle|\, 0, \boldsymbol{A}_k^{-1}\right). \tag{5.4}$$

where $\boldsymbol{A}_k$ is a diagonal matrix of variances $a_k(p)$ at each pixel p. Chantas *et al.*[16] model the distribution of $a_k(p)$ as a Gamma distribution, which leads to a heavy-tailed marginal distribution for $\hat{\boldsymbol{f}}_k$. A similar approach has been used by Levin *et al.*[50], but they impose $\boldsymbol{A}_k = \alpha \boldsymbol{I}$. The assumption in this method is that $\boldsymbol{A}_k$ is a diagonal matrix of unknown values which makes our marginalisation tractable. We write $\boldsymbol{A} = \{\boldsymbol{A}_1 \cdots \boldsymbol{A}_P\}$.

In general, as anticipated in Section 4.1, the complete inference problem may be seen as estimating $\boldsymbol{d}$, $\hat{\boldsymbol{f}}$, and $\boldsymbol{A}$ from the observations $\hat{\boldsymbol{g}} = [\hat{\boldsymbol{g}}_1^T, \cdots, \hat{\boldsymbol{g}}_P^T]^T$. Since the interest here is in depth estimation alone, the following problem is instead considered

$$\boldsymbol{d}^* = \underset{\boldsymbol{d}}{\operatorname{argmax}}\, p\left(\boldsymbol{d} \mid \hat{\boldsymbol{g}}, \boldsymbol{A}\right) \tag{5.5}$$

$$= \underset{\boldsymbol{d}}{\operatorname{argmax}}\, p\left(\hat{\boldsymbol{g}} \mid \boldsymbol{d}, \boldsymbol{A}\right) p\left(\boldsymbol{d}\right). \tag{5.6}$$

In this case, *shape from coded aperture* refers to the problem of reconstructing the projected depth map $\boldsymbol{d}$ given the set of observed filtered images $\hat{\boldsymbol{g}}$, described in equation (5.5). In the next section the marginal likelihood in equation (5.6) is obtained.

5.2 Bayesian Depth Inference

This section describes how to estimate the depth map directly from the observations without explicit estimation of the texture.

5.2.1 Marginalisation

To begin the analysis, $\hat{\boldsymbol{f}}_k$ is marginalised as follows:

$$p\left(\hat{\boldsymbol{g}}_k \mid \boldsymbol{d}, \boldsymbol{A}_k\right) = \int p\left(\hat{\boldsymbol{g}}_k, \hat{\boldsymbol{f}}_k \mid \boldsymbol{d}, \boldsymbol{A}_k\right) \mathrm{d}\hat{\boldsymbol{f}}_k \tag{5.7}$$

$$= \int p\left(\hat{\boldsymbol{g}}_k \mid \hat{\boldsymbol{f}}_k, \boldsymbol{d}\right) p\left(\hat{\boldsymbol{f}}_k \mid \boldsymbol{A}_k\right) \mathrm{d}\hat{\boldsymbol{f}}_k \tag{5.8}$$

$$= \mathcal{N}\left(\hat{\boldsymbol{g}}_k \mid \boldsymbol{\mu}_k(\boldsymbol{A}_k), \boldsymbol{\Sigma}_k(\boldsymbol{A}_k)\right) \tag{5.9}$$

where[1]

$$\boldsymbol{\mu}_k(\boldsymbol{A}_k) = 0 \tag{5.10}$$

$$\boldsymbol{\Sigma}_k(\boldsymbol{A}_k) = \boldsymbol{H}_{\boldsymbol{d}} \boldsymbol{A}_k^{-1} \boldsymbol{H}_{\boldsymbol{d}}^T + \boldsymbol{C}_k \sigma^2 \boldsymbol{I}. \tag{5.11}$$

This integration is achieved by applying the Gaussian integral.[2] One could estimate $\boldsymbol{A}_k$ and use the definition of $\boldsymbol{\Sigma}_k$ to evaluate the likelihood in equation (5.9). In this case, for simplicity $\boldsymbol{\Sigma}_k$ is estimated directly from the data. This becomes tractable due to (i) the fact that equation (5.9) is Gaussian, which allows us to work with local conditional distributions (Section 5.2.2) and (ii) the structure of $\boldsymbol{\Sigma}_k$ (Section 5.2.3).

5.2.2 Local Factorisation of $\boldsymbol{\Sigma}_k$

To work locally, the Markov Random Field (MRF) principle of conditional independence may be applied, if it can be demonstrated that the pixel $\mathbf{p}$ only depends on certain neighbours in a given small region N_p:

$$p\left(\hat{\boldsymbol{g}}_k[\mathbf{p}] \mid \hat{\boldsymbol{g}}_k[\backslash \mathbf{p}], \boldsymbol{d}\right) = p\left(\hat{\boldsymbol{g}}_k[\mathbf{p}] \,\middle|\, \hat{\boldsymbol{g}}_k[N_p], \boldsymbol{d}\right). \tag{5.12}$$

[1] The mean is given by $\boldsymbol{\mu}_k(\boldsymbol{A}_k) = \boldsymbol{\Sigma}_k^{-1}\left(\sigma^{-2}\boldsymbol{I} + \boldsymbol{H}_{\boldsymbol{d}}^{-T}\boldsymbol{A}_k\boldsymbol{H}_{\boldsymbol{d}}^{-1}\right)\sigma^{-2}\boldsymbol{H}_{\boldsymbol{d}}^{-1}\boldsymbol{A}_k\boldsymbol{\mu}_{\hat{f}}$ where $\boldsymbol{\mu}_{\hat{f}} = 0$ in our image prior model.

[2] Due to normalisation of the Gaussian distribution, we have in general that $\int\limits_{\mathbb{R}^{P\times 1}} \cdots \int \exp\left[-\frac{1}{2}\left(\boldsymbol{x}^T\boldsymbol{\Gamma}\boldsymbol{x} - 2\boldsymbol{\beta}^T\boldsymbol{x} + \alpha\right)\right] \mathrm{d}\boldsymbol{x} = \frac{(2\pi)^{P/2}}{\sqrt{\det|\boldsymbol{\Gamma}|}} \exp\left[-\frac{1}{2}\left(\alpha - \boldsymbol{\beta}^T\boldsymbol{\Gamma}^{-1}\boldsymbol{\beta}\right)\right]$.

In other words, rather than considering all other pixels $\hat{\boldsymbol{g}}_k[\backslash \mathbf{p}]$ in the above expressions, one can just work with $\hat{\boldsymbol{g}}_k[N_p]$. This will be shown in Section 5.2.3.

Since $\hat{\boldsymbol{g}}_k$ is Gaussian, the conditional distribution of one pixel $\hat{\boldsymbol{g}}_k[\mathbf{p}]$ in the image given the rest $\hat{\boldsymbol{g}}_k[\backslash \mathbf{p}]$ is also Gaussian, with PDF

$$p\left(\hat{\boldsymbol{g}}[\mathbf{p}] \,|\, \hat{\boldsymbol{g}}[\backslash p], \boldsymbol{d}\right) = \mathcal{N}\left(\hat{\boldsymbol{g}}[\mathbf{p}] \middle| \nu_{\mathbf{p}|\backslash \mathbf{p}}, \Gamma_{p|\backslash p}\right) \tag{5.13}$$

$$= \mathcal{N}\left(\hat{\boldsymbol{g}}[\mathbf{p}] \middle| \nu_{\mathbf{p}|N_p}, \Gamma_{\mathbf{p}|N_p}\right), \tag{5.14}$$

with

$$\nu_{\mathbf{p}|N_p} = \boldsymbol{\mu}[\mathbf{p}] + \boldsymbol{\Sigma}[\mathbf{p}, N_p]\boldsymbol{\Sigma}[N_p, N_p]^{-1}\left(\hat{\boldsymbol{g}}[N_p] - \boldsymbol{\mu}[N_p]\right) \tag{5.15}$$

$$\Gamma_{\mathbf{p}|N_p} = \boldsymbol{\Sigma}[\mathbf{p}, \mathbf{p}] - \boldsymbol{\Sigma}[\mathbf{p}, N_p]\boldsymbol{\Sigma}[N_p, N_p]^{-1}\boldsymbol{\Sigma}[N_p, \mathbf{p}], \tag{5.16}$$

and $\boldsymbol{\mu}[\mathbf{p}]$ and $\boldsymbol{\mu}[N_p]$ become zero from the assumption described in Section 5.1.1. The subscripts $(_k)$ are assumed but omitted for clarity and indices inside brackets address rows and columns of $\boldsymbol{\Sigma}_k$, such that the following structure contains all non-zero elements pertaining to the pixel $\mathbf{p}$

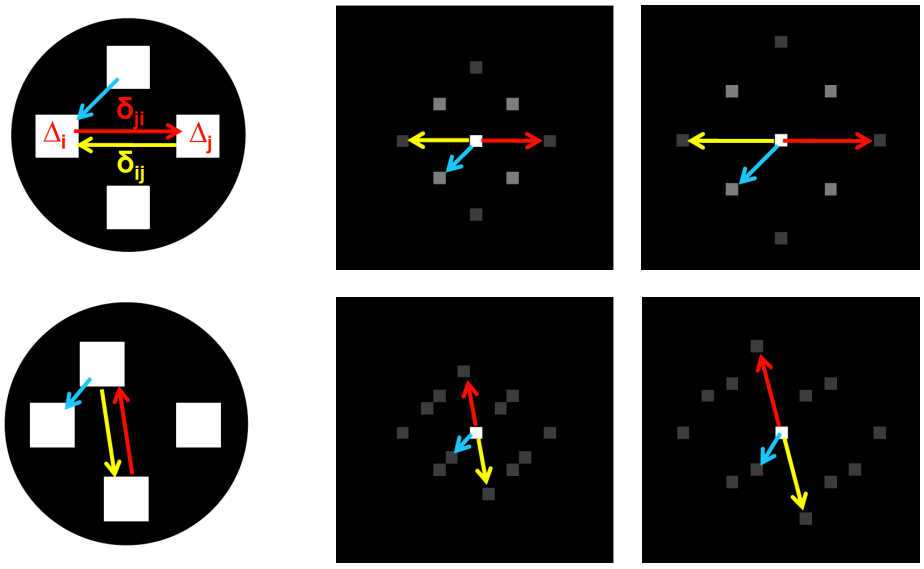
$$\left[\begin{array}{c|c} \boldsymbol{\Sigma}[\mathbf{p}, \mathbf{p}] & \boldsymbol{\Sigma}[\mathbf{p}, N_p] \\ \hline \boldsymbol{\Sigma}[N_p, \mathbf{p}] & \boldsymbol{\Sigma}[N_p, N_p] \end{array}\right] \tag{5.17}$$

where $\boldsymbol{\Sigma}[\mathbf{p}, N_p]$ is of size $1 \times |N_p|$ and $\boldsymbol{\Sigma}[N_p, \mathbf{p}] = \Sigma[\mathbf{p}, N_p]^T$.

5.2.3 Structure of the Local Neighbourhood in $\boldsymbol{\Sigma}_k$

Since $\boldsymbol{A}_k$ is diagonal, the neighbourhood structure N_p only depends on the offsets in $\boldsymbol{H_d}$. In fact, the contribution at a pixel $\mathbf{p}$, generated by $\boldsymbol{H_d}\boldsymbol{A}_k^{-1}\boldsymbol{H_d}^T$ for a given distance d, is limited to a neighborhood, whose structure can be defined as

$$N_p = \left\{\boldsymbol{p} + \boldsymbol{\delta}_{ij} d \;|\, i \neq j \wedge i, j \in \mathcal{M}\right\} \tag{5.18}$$



(a) mask (b) depth d_1 (c) depth d_2

Figure 5.1: Structure of N_p for $d_1 < d_2$. Examples of the structure of N_p in a coded image for a 4-hole symmetric aperture mask (top row) and a 4-hole asymmetric mask (bottom). We show the neighborhood for two depths, with depth d_1 closer to the focal plane than depth d_2. Colored arrows may help the reader to link the structure of the mask to the pixels belonging to N_p, as defined in equation (5.18).

where $\boldsymbol{\delta}_{ij} = (\boldsymbol{\Delta}_i - \boldsymbol{\Delta}_j)$ is a vector that represents the distance between the aperture i and the aperture j in the mask $\mathcal{M}$. In Figure 5.1 the same terminology is used to illustrate how the neighborhood N_p is related to the shape of the aperture mask and how its structure changes with the distance d. The bright point at the center of each image indicates the pixel $\mathbf{p}$ and the surrounding points represent the neighbourhood N_p.

The number of elements in N_p is given by

$$|N_p| = \frac{N!}{(N-2)!} = N(N-1), \tag{5.19}$$

which indicates that the amount of computations of our algorithm increases with the number of apertures N in the mask. This is also illustrated in Figure 5.2, where the neighborhood N_p is shown for different aperture masks.

Since it has been verified that the pixel $\mathbf{p}$ only depends on a small finite number of

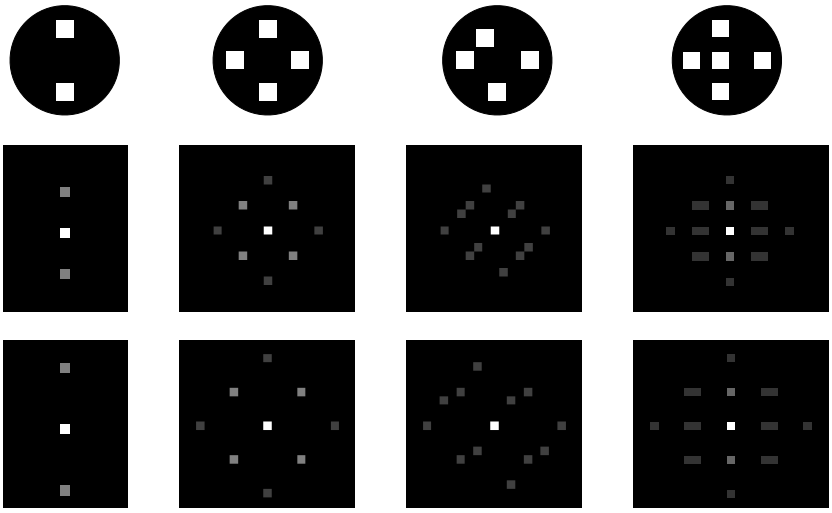


Figure 5.2: Neighborhood N_p for different masks. The neighborhood of the aperture masks in the first row is illustrated for two depths, d_1 (central row) and $d_2 > d_1$ (bottom row). For the second and for the last mask, some of the neighbours in N_p are counted more than once (brighter color).

neighbours N_p, the MRF principle of conditional independence can be applied:

$$p\left(\hat{\boldsymbol{g}} \mid \boldsymbol{A}, \boldsymbol{d}\right) = \prod_{p=1\ldots M} p\left(\hat{\boldsymbol{g}}_k(\mathbf{p}) \mid \hat{\boldsymbol{g}}_k(N_p), \boldsymbol{d}\right). \tag{5.20}$$

Due to just one observation of the image being available, the ergodicity assumption of local stationarity is employed, that is a local window can be used to estimate the required statistics at each point in the image.

5.2.4 MAP Estimation of Depth Map

Given the local estimates of the image mean and variance conditional on each possible depth (assuming a discrete set of depth values corresponding to integer disparities), one can consider maximising the posterior for $\boldsymbol{d}$ in equation (5.6). Due to the independence of the filtered observations [16],

$$p\left(\hat{\boldsymbol{g}} \mid \boldsymbol{A}, \boldsymbol{d}\right) = \prod_{k=1\ldots P} p\left(\hat{\boldsymbol{g}}_k \mid \boldsymbol{A}_k, \boldsymbol{d}\right) . \tag{5.21}$$

The prior $p(\boldsymbol{d})$ is defined as the penalty term on the gradients of the depth map in the L_1 norm (Gibbs distribution). The next step is to take the negative logarithm of the likelihood in equation (5.6), apply the MRF principle in equation (5.20), and successively equation (5.14); this yields

$$\boldsymbol{d}^* = \underset{\boldsymbol{d}}{\operatorname{argmin}} \left(E_{data}(\boldsymbol{d}) + E_{sm}(\boldsymbol{d}) \right) \tag{5.22}$$

with

$$E_{data}(\boldsymbol{d}) = \frac{1}{2} \sum_k \sum_{\mathbf{p}} \left[(\hat{\boldsymbol{g}}_k(\mathbf{p}) - \nu_{\mathbf{p}|N_p})^T \Gamma^{-1}_{\mathbf{p}|N_p} (\hat{\boldsymbol{g}}_k(\mathbf{p}) - \nu_{\mathbf{p}|N_p}) + \log(2\pi \det |\Gamma_{\mathbf{p}|N_p}|) \right] \tag{5.23}$$

$$E_{sm}(\boldsymbol{d}) = -\log p(\boldsymbol{d}) = \sum_{\mathbf{p}, \{\mathbf{q} \in V_{\mathbf{p}}\}} \min(|d_{\mathbf{p}} - d_{\mathbf{q}}|, T), \tag{5.24}$$

where V_p is the neighborhood of a pixel $\mathbf{p}$ and T is a constant. Thus E_{sm} penalizes differences in the depths of neighboring pixels. The inference procedure consists of minimising the energy given by equation (5.22) via Graph-Cuts [45]. In the implementation presented in thesis the number of operators $\boldsymbol{C}_k$ is $P = 2$, and they correspond to discrete horizontal and vertical derivatives.

5.3 Results and Discussion

5.3.1 Performance

The proposed algorithm has been compared with five methods previously proposed for coded aperture images, on different types of aperture. Since the computational cost of the algorithm is rapidly increasing with the number of apertures in the mask (as described in equation (5.19)), only 3 simple patters are considered: 2-hole, 3-hole, and 4-hole masks. Coded images have been synthetically simulated by placing a plane of random texture at 33 different known depths. The coded images are then given as input to the five algorithms and the estimated depths, $\boldsymbol{d}$, are compared with the ground-truth $\hat{\boldsymbol{d}}$. The distance between the two depth profiles represents the error

Methods	**Masks** (image noise level $\sigma = 0.0001$)		
	2-hole	3-hole	4-hole
Lucy-Richardson	14.2	15.4	13.9
Regular filters	11.9	14.9	13.9
Wiener filters	17.0	17.0	15.1
Gaussian priors	12.1	15.2	13.7
Levin *et al.*	13.9	15.2	13.7
Our method	**8.7**	**9.3**	**8.7**

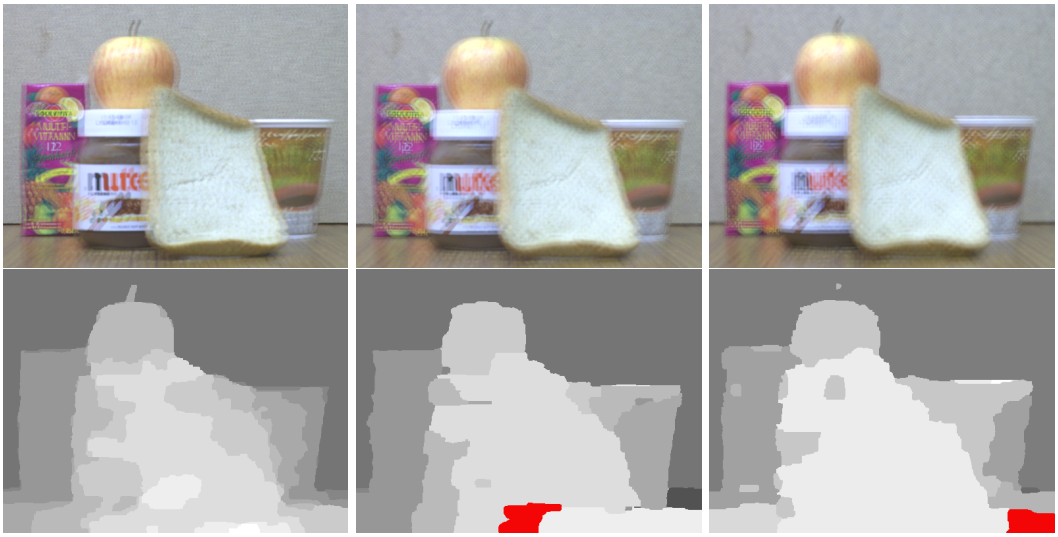
Table 5.1: Performance comparison (mean error).

of the reconstruction: $ERR = |\boldsymbol{d} - \hat{\boldsymbol{d}}|$. The mean error reported in Table 5.1 is the average of all the errors for a given method and a given aperture mask (occlusions are not considered). SNR is taken into account by considering the amount of light that goes through each aperture. Since the proposed algorithm does not restore the sharp image, its computational time is very low for the types of masks analysed here: it takes about 1 minute (in a Pentium Core2Duo 3.00GHz) to compute the depth map of a coded image of size 640×480 taken with a 2-hole mask, such as the datasets shown in Figure 5.5.

5.3.2 Real Data

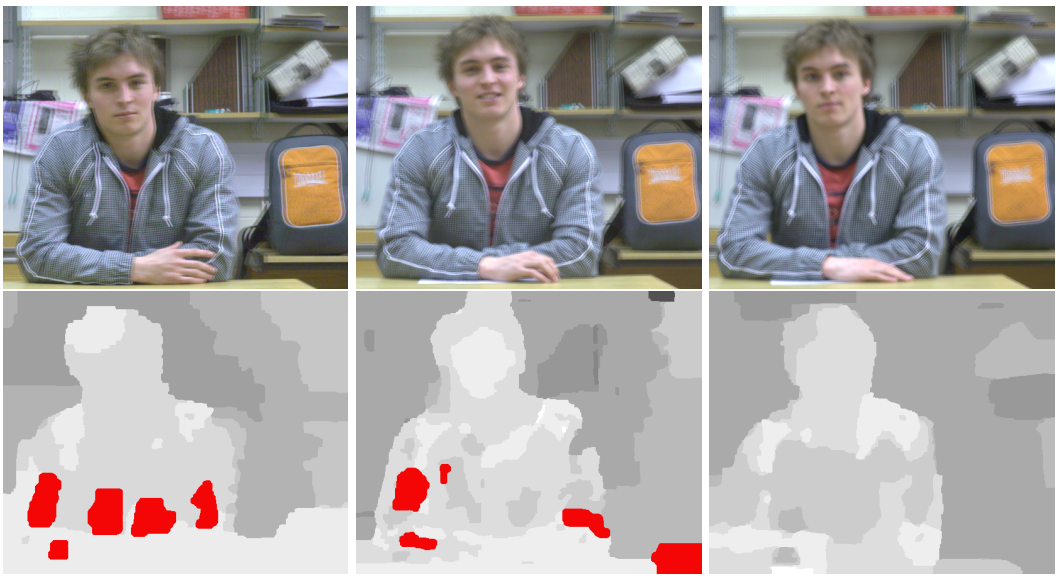
Coded aperture images were obtained by inserting a mask into a 50mm $f/1.4$ lens mounted on a Canon EOS-5D DSLR. The exposure time was set to 40ms (ISO 500) for images captured with the 2-hole mask, 33ms (ISO 400) with the 3-hole mask, and 20ms (ISO 400) for the 4-hole mask. Each aperture in the mask is a 4×4 mm square, and the distance between the centers of the holes is about 13 mm.

The method was applied to two different kinds of scenario to show how it performs with different ranges of depths and changes of mask. In order to maximise the disparities, the focal plane of the camera lens is set to be just after the object of interest (Figure 5.5(a) and Figure 5.3(a, c)) or just before them (Figure 5.4(d, f)). Figure 5.3(a-c) displays a scene with several objects placed at distances between 80cm and 120cm from the camera lens, while Figure 5.3(d, f) represents a scene with a wider range of



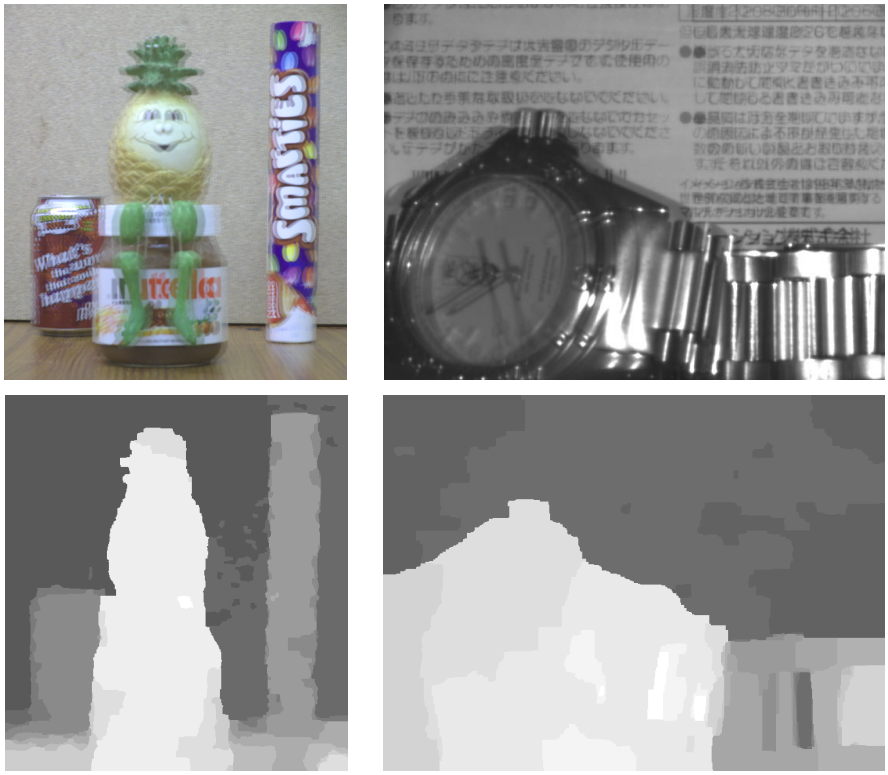
(a) (b) (c)

Figure 5.3: Real data - Snacks dataset.. Images given as input to our algorithm (top) and their relative depth map (bottom). The two scenes has been both captured with 3 different aperture masks: 2-hole (a), 3-hole (b), and 4-hole (c). Red colour represents areas where depth has not been estimated.



(a) (b) (c)

Figure 5.4: Real data - Person dataset. Images given as input to our algorithm (top) and their relative depth map (bottom). The two scenes has been both captured with 3 different aperture masks: 2-hole (a), 3-hole (b), and 4-hole (c). Red colour represents areas where depth has not been estimated.



(a) (b)

Figure 5.5: Depth estimation with the 2-hole mask. The input images at the top have been capture with a 2-hole aperture mask. The dataset in (a) has been captured with our coded aperture camera, while the image in (b) has been extracted from the paper of Hiura and Matsuyama [39].

depths (from 200cm to 350cm). One can notice from the estimated depth maps that, when the number of the apertures in the mask is increased, we loose details but we solve some ambiguities due to occlusion or repeating texture which are present in images captured with masks with 2 or 3 apertures. Figure 5.5 shows two a depth maps obtained from coded aperture images capture with a 2-hole mask. The dataset in Figure 5.5(a) has been captured with the focal plane set at 120cm and the objects placed in a range of 50cm. The result obtained in Figure 5.5(b) is very interesting since the dataset has not been captured with our coded aperture camera, but instead extracted from the paper of Hiura and Matsuyama [39], who uses the same type of aperture mask.

5.4 Summary

The chapter has presented an analysis and an algorithm to solve shape from coded aperture, without the need of recovering the sharp image. The novel depth inference proposed here has higher performance then previous methods based on a single coded image as input. Priors on the scene texture and depth map are also introduced to solve ambiguities in the solution.

This page has been left intentionally blank.

Chapter 6

Blur Estimation and Image Deblurring for General Patterns

You can always change you plan,
but only if you have one.
Randy Pausch [1960-2008]

The previous chapter described how to estimate the depth from a single coded image bypassing the deblurring procedure. This chapter presents a more general approach to solve the same initial problem: depth and all-in-focus image reconstruction from a single coded image. The method described in this chapter has two main advantages over the previous one: 1) it is not limited to a set of masks, and 2) no assumptions are made on the statistics of the sharp image, but instead it is learned automatically from a set of natural images. Moreover, the novel algorithm presented here is computationally efficient and it achieves state-of-the-art performance in terms of depth and image reconstruction with coded aperture cameras (Section 6.3).

Since the depth of an object is related to its blur size (and the relationship can be obtained with the calibration procedure described in Section 3.4), the estimation is restricted to the blur size.

6.1 Single Image Blind Deconvolution

Blind deconvolution from a single image is a very challenging problem: One needs to recover more unknowns than the available observations. This challenge will be illustrated in the next section, where the image formation model of a coded image will be recalled. To make the problem feasible and well-behaved, one can introduce additional constraints on the solution. In particular, the higher-order statistics of sharp texture are constrained (sec. 6.1.2) and the blur scale is imposed to be piecewise smooth across the image pixels (sec. 6.1.3).

6.1.1 Problem Statement

Recalled below is the image formation model formulated in equation (3.27)

$$\boldsymbol{g} = \boldsymbol{H_d f} + \boldsymbol{w} \ , \tag{6.1}$$

where the i-th column of $\boldsymbol{H_d}$ is an image, rearranged as a vector, of the coded blur with scale $\boldsymbol{d}_i$ generated by the i-th pixel of $\boldsymbol{f}$. Given the blurred image $\boldsymbol{g}$, to recover the unknown sharp image $\boldsymbol{f}$ one needs to recover also the blur scale at each pixel $\boldsymbol{d}$. As described in Section 4.1, the problem can be formulated in a Bayesian framework as

$$\begin{aligned} \boldsymbol{d}^*, \boldsymbol{f}^* &= \underset{\boldsymbol{d},\boldsymbol{f}}{\operatorname{argmax}}\, p\left(\boldsymbol{d}, \boldsymbol{f} \mid \boldsymbol{g}\right) \\ &= \underset{\boldsymbol{d},\boldsymbol{f}}{\operatorname{argmax}}\, p\left(\boldsymbol{g} \mid \boldsymbol{f}, \boldsymbol{d}\right) p\left(\boldsymbol{f}\right) p\left(\boldsymbol{d}\right) \ , \end{aligned} \tag{6.2}$$

where the prior on the sharp image $p\left(\boldsymbol{f}\right)$ and on the blur scale (or depth) $p\left(\boldsymbol{d}\right)$ have to be defined in order to obtain a unique reliable solution. Both definitions are based on the observation that, typically, one expects the unknown sharp image and blur scale map to have some regularity. For instance, both sharp textures and blur scale maps are not likely to look like noise. The next two sections will present and illustrate our

sharp image and blur scale priors.

6.1.2 Sharp Image Prior

Images of the real world exhibit statistical regularities that have been studied intensively in the past 20 years and have been linked to the human visual system and its evolution [73]. For the purpose of image deblurring, the most important aspect of this study is that natural images form a much smaller subset of all possible images. In general, the characterization of the statistical properties of natural images is done by applying a given transform, typically related to a component of human vision. Among the most common statistics used in image processing are the second order statistics, *i.e.*, relations between pairs of pixels. For instance, this category includes the distributions of image gradients [80, 40].

However, a more accurate account of the image structure can be captured with high-order statistics, *i.e.*, relations between several pixels. In this work this general case is considered, but the relations are restricted to linear ones of the form

$$\boldsymbol{\Sigma} \boldsymbol{f} \simeq 0 \tag{6.3}$$

where $\boldsymbol{\Sigma}$ is a rectangular matrix. Equation (6.3) implies that all sharp images live approximately on a subspace. Despite their crude simplicity, these linear constraints allow for some flexibility. For example, the case of second-order statistics results in rows of $\boldsymbol{\Sigma}$ with only two nonzero values. Also, by designing $\boldsymbol{\Sigma}$ one can selectively apply the constraints only on some of the pixels. Another example is to choose each row of $\boldsymbol{\Sigma}$ as a Haar feature applied to some pixels. Notice that this approach does not make any of these choices. Rather, $\boldsymbol{\Sigma}$ is estimated directly from natural images.

Natural image statistics, such as gradients, typically exhibit a peaked distribution. However, performing inference on such distributions results in minimizations of non convex functionals for which there are probably not optimal algorithms. Furthermore, our interest here is to simplify the optimization task as much as possible to gain in

computational efficiency. To this end, one can enforce the linear relation above by minimizing the convex cost

$$\|\boldsymbol{\Sigma}\boldsymbol{f}\|_2^2. \tag{6.4}$$

As there is no analytical expression for $\boldsymbol{\Sigma}$ that satisfies equation (6.3), it has to be learned directly from the data. This step is necessary only when performing the deblurring procedure given the estimated blur, as will be explained later. Instead, when estimating the blur scale, the method allows us to use $\boldsymbol{\Sigma}$ implicitly, *i.e.*, without ever recovering it.

6.1.3 Blur Scale Prior

The statistics of range images can be characterized with an approach similar to that for optical images [41]. The study in [41] verified the random collage model, *i.e.*, that a scene is a collection of piecewise constant surfaces. This has been observed in the distributions of Haar filter responses on the logarithm of the range data, which showed strong cusps in the isoprobability contours. Unfortunately, a prior following these distributions faithfully would result in non convex energy minimization. A practical convex solution to enforce the piecewise constant model, is to use total variation [81]. Common choices are the isotropic and anisotropic total variation. In our algorithm the latter is implemented. One has to minimize $\|\nabla \boldsymbol{d}\|_1$, *i.e.*, the sum of the absolute value of the components of the gradient of $\boldsymbol{d}$.

6.2 Blur Scale Identification and Image Deblurring

When the image model introduced in sec. 6.1.1 is combined with the priors in sec. 6.1.2 and 6.1.3 one can formulate the following energy minimization problem

$$\boldsymbol{d}^*, \boldsymbol{f}^* = \underset{\boldsymbol{d},\boldsymbol{f}}{\operatorname{argmin}} \|\boldsymbol{g} - \boldsymbol{H}_{\boldsymbol{d}}\boldsymbol{f}\|_2^2 + \alpha\|\boldsymbol{\Sigma}\boldsymbol{f}\|_2^2 + \beta\|\nabla \boldsymbol{d}\|_1, \tag{6.5}$$

where the parameters $\alpha, \beta > 0$ determine the amount of regularization for texture and blur scale respectively. Notice that the formulation above is common to many approaches including, in particular, [50]. Our approach, however, in addition to using a more accurate blur matrix $\boldsymbol{H_d}$, considers different priors and a different depth identification procedure.

Our next step is to notice that, given $\boldsymbol{d}$, the proposed cost is simply a least-squares problem in the unknown sharp texture $\boldsymbol{f}$. Hence, it is possible to compute $\boldsymbol{f}$ in closed-form and plug it back in the cost functional. The result is a much simpler problem to solve. All the steps are summarised in the following Theorem:

Theorem 6.2.1 *The set of extrema of the minimization* (6.5) *coincides with the set of extrema of the minimization*

$$\begin{cases} \boldsymbol{d}^* &= \underset{\boldsymbol{d}}{\operatorname{argmin}} \|\boldsymbol{H}_{\boldsymbol{d}}^{\perp}\boldsymbol{g}\|_2^2 + \beta\|\nabla\boldsymbol{d}\|_1 \\ \boldsymbol{f}^* &= \left(\alpha\boldsymbol{\Sigma}^T\boldsymbol{\Sigma} + \boldsymbol{H}_{\boldsymbol{d}^*}^T\boldsymbol{H}_{\boldsymbol{d}^*}\right)^{-1}\boldsymbol{H}_{\boldsymbol{d}^*}^T\boldsymbol{g} \end{cases} \tag{6.6}$$

where

$$\boldsymbol{H}_{\boldsymbol{d}}^{\perp} \doteq \boldsymbol{I} - \boldsymbol{H}_{\boldsymbol{d}}\left(\alpha\boldsymbol{\Sigma}^T\boldsymbol{\Sigma} + \boldsymbol{H}_{\boldsymbol{d}}^T\boldsymbol{H}_{\boldsymbol{d}}\right)^{-1}\boldsymbol{H}_{\boldsymbol{d}}^T\,, \tag{6.7}$$

and $\boldsymbol{I}$ is the identity matrix.

Proof To prove the theorem we rewrite the least squares problem in $\boldsymbol{f}$ as

$$\|\boldsymbol{H}_{\boldsymbol{d}}\boldsymbol{f} - \boldsymbol{g}\|_2^2 + \alpha\|\boldsymbol{\Sigma}\boldsymbol{f}\|_2^2 = \left\| \begin{bmatrix} \boldsymbol{H}_{\boldsymbol{d}} \\ \sqrt{\alpha}\boldsymbol{\Sigma} \end{bmatrix} \boldsymbol{f} - \begin{bmatrix} \boldsymbol{g} \\ 0 \end{bmatrix} \right\|_2^2 = \|\bar{\boldsymbol{H}}_{\boldsymbol{d}}\boldsymbol{f} - \bar{\boldsymbol{g}}\|_2^2 \tag{6.8}$$

where it is defined $\bar{\boldsymbol{H}}_{\boldsymbol{d}} = \begin{bmatrix}\boldsymbol{H}_{\boldsymbol{d}}^T & \sqrt{\alpha}\boldsymbol{\Sigma}^T\end{bmatrix}^T$ and $\bar{\boldsymbol{g}} = \begin{bmatrix}\boldsymbol{g}^T & 0^T\end{bmatrix}^T$. Then the solution in $\boldsymbol{f}$ can be written as $\boldsymbol{f}^* = \left(\bar{\boldsymbol{H}}_{\boldsymbol{d}}^T\bar{\boldsymbol{H}}_{\boldsymbol{d}}\right)^{-1}\bar{\boldsymbol{H}}_{\boldsymbol{d}}^T\bar{\boldsymbol{g}}$. By substituting the solution for $\boldsymbol{f}$ back in the least squares problem,

$$\|\boldsymbol{H}_{\boldsymbol{d}}\boldsymbol{f} - \boldsymbol{g}\|_2^2 + \alpha\|\boldsymbol{\Sigma}\boldsymbol{f}\|_2^2 = \|\bar{\boldsymbol{H}}_{\boldsymbol{d}}^{\perp}\bar{\boldsymbol{g}}\|_2^2 \tag{6.9}$$

where

$$\bar{\boldsymbol{H}}_{\boldsymbol{d}}^{\perp} = \boldsymbol{I} - \bar{\boldsymbol{H}}_{\boldsymbol{d}} \left(\bar{\boldsymbol{H}}_{\boldsymbol{d}}^{T} \bar{\boldsymbol{H}}_{\boldsymbol{d}} \right)^{-1} \bar{\boldsymbol{H}}_{\boldsymbol{d}}^{T} \tag{6.10}$$

$$= \begin{bmatrix} A & B \\ C & D \end{bmatrix} \tag{6.11}$$

with

$$A = \boldsymbol{I} - \boldsymbol{H}_{\boldsymbol{d}} \left(\boldsymbol{H}_{\boldsymbol{d}}^{T} \boldsymbol{H}_{\boldsymbol{d}} + \alpha \boldsymbol{\Sigma}^{T} \boldsymbol{\Sigma} \right)^{-1} \boldsymbol{H}_{\boldsymbol{d}}^{T} \doteq \boldsymbol{H}_{\boldsymbol{d}}^{\perp} \tag{6.12}$$

$$B = -\boldsymbol{H}_{\boldsymbol{d}} \left(\boldsymbol{H}_{\boldsymbol{d}}^{T} \boldsymbol{H}_{\boldsymbol{d}} + \alpha \boldsymbol{\Sigma}^{T} \boldsymbol{\Sigma} \right)^{-1} \sqrt{\alpha} \boldsymbol{\Sigma}^{T} \tag{6.13}$$

$$C = -\sqrt{\alpha} \boldsymbol{\Sigma} \left(\boldsymbol{H}_{\boldsymbol{d}}^{T} \boldsymbol{H}_{\boldsymbol{d}} + \alpha \boldsymbol{\Sigma}^{T} \boldsymbol{\Sigma} \right)^{-1} \boldsymbol{H}_{\boldsymbol{d}}^{T} \tag{6.14}$$

$$D = \boldsymbol{I} - \sqrt{\alpha} \boldsymbol{\Sigma} \left(\boldsymbol{H}_{\boldsymbol{d}}^{T} \boldsymbol{H}_{\boldsymbol{d}} + \alpha \boldsymbol{\Sigma}^{T} \boldsymbol{\Sigma} \right)^{-1} \sqrt{\alpha} \boldsymbol{\Sigma}^{T} \tag{6.15}$$

The step above is necessary to fully exploit the properties of $\bar{\boldsymbol{H}}_{\boldsymbol{d}}^{\perp}$. $\bar{\boldsymbol{H}}_{\boldsymbol{d}}^{\perp}$ is a *symmetric* matrix (i.e, $(\bar{\boldsymbol{H}}_{\boldsymbol{d}}^{\perp})^{T} = \bar{\boldsymbol{H}}_{\boldsymbol{d}}^{\perp}$) and is also *idempotent* (i.e, $\bar{\boldsymbol{H}}_{\boldsymbol{d}}^{\perp} = (\bar{\boldsymbol{H}}_{\boldsymbol{d}}^{\perp})^{2}$). By applying the above properties one can write the argument of the first term of the cost in equation (6.6) as

$$\bar{\boldsymbol{g}}^{T} \bar{\boldsymbol{H}}_{\boldsymbol{d}}^{\perp} \bar{\boldsymbol{g}} = \bar{\boldsymbol{g}}^{T} (\bar{\boldsymbol{H}}_{\boldsymbol{d}}^{\perp})^{T} \bar{\boldsymbol{H}}_{\boldsymbol{d}}^{\perp} \bar{\boldsymbol{g}} = ||\bar{\boldsymbol{H}}_{\boldsymbol{d}}^{\perp} \bar{\boldsymbol{g}}||_{2}^{2} \,. \tag{6.16}$$

By using the matrix structure in equation (6.11), equation (6.16) can be express as

$$\|\bar{\boldsymbol{H}}_{\boldsymbol{d}}^{\perp} \bar{\boldsymbol{g}}\|_{2}^{2} = \begin{bmatrix} \boldsymbol{g}^{T} & 0^{T} \end{bmatrix} \begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} \boldsymbol{g} \\ 0 \end{bmatrix} = \boldsymbol{g}^{T} A \boldsymbol{g} = \|\boldsymbol{H}_{\boldsymbol{d}}^{\perp} \boldsymbol{g}\|_{2}^{2} \,. \tag{6.17}$$

Therefore one can use $\bar{\boldsymbol{H}}_{\boldsymbol{d}}^{\perp}$ rather than $\boldsymbol{H}_{\boldsymbol{d}}^{\perp}$ and $\bar{\boldsymbol{g}}$ rather than $\boldsymbol{g}$ in the minimization problem (6.6) without affecting the solution. The rest of the proof then assumes that the energy in equation (6.6) is based on $\|\bar{\boldsymbol{H}}_{\boldsymbol{d}}^{\perp} \bar{\boldsymbol{g}}\|_{2}^{2}$.

Moreover, from the definition of $\bar{\boldsymbol{H}}_{\boldsymbol{d}}^{\perp}$ it is known that

$$\begin{aligned}\bar{\boldsymbol{H}}_{\boldsymbol{d}}^{\perp} &\doteq \boldsymbol{I} - \bar{\boldsymbol{H}}_{\boldsymbol{d}}(\bar{\boldsymbol{H}}_{\boldsymbol{d}}^{T}\bar{\boldsymbol{H}}_{\boldsymbol{d}})^{-1}\bar{\boldsymbol{H}}_{\boldsymbol{d}}^{T} \\ &= \boldsymbol{I} - \bar{\boldsymbol{H}}_{\boldsymbol{d}}\bar{\boldsymbol{H}}_{\boldsymbol{d}}^{\dagger}\,, \end{aligned} \tag{6.18}$$

where $\bar{\boldsymbol{H}}_{\boldsymbol{d}}^{\dagger}$ is the pseudo-inverse of $\bar{\boldsymbol{H}}_{\boldsymbol{d}}$ [33]. Thus, the necessary conditions for an extremum of equation (6.6) become

$$\begin{cases} \left(\bar{\boldsymbol{g}} - \bar{\boldsymbol{H}}_{\boldsymbol{d}}\bar{\boldsymbol{H}}_{\boldsymbol{d}}^{\dagger}\bar{\boldsymbol{g}}\right)^{T}\left(\nabla\bar{\boldsymbol{H}}_{\boldsymbol{d}}\bar{\boldsymbol{H}}_{\boldsymbol{d}}^{\dagger} + \bar{\boldsymbol{H}}_{\boldsymbol{d}}\nabla\bar{\boldsymbol{H}}_{\boldsymbol{d}}^{\dagger}\right)\bar{\boldsymbol{g}} &= \nabla\cdot\dfrac{\nabla\boldsymbol{d}}{\|\nabla\boldsymbol{d}\|_1} \\ \boldsymbol{f} &= \bar{\boldsymbol{H}}_{\boldsymbol{d}}^{\dagger}\bar{\boldsymbol{g}}. \end{cases} \tag{6.19}$$

where $\nabla\bar{\boldsymbol{H}}_{\boldsymbol{d}}$ is the gradient of $\bar{\boldsymbol{H}}_{\boldsymbol{d}}$ with respect to $\boldsymbol{d}$, and the right hand side of the first equation is the gradient of $\|\nabla\boldsymbol{d}\|_1$ with respect to $\boldsymbol{d}$. Similarly, the necessary conditions for equation (6.5) are

$$\begin{cases} \left(\bar{\boldsymbol{g}} - \bar{\boldsymbol{H}}_{\boldsymbol{d}}\boldsymbol{f}\right)^{T}\nabla\bar{\boldsymbol{H}}_{\boldsymbol{d}}\boldsymbol{f} &= \nabla\cdot\dfrac{\nabla\boldsymbol{d}}{\|\nabla\boldsymbol{d}\|_1} \\ \bar{\boldsymbol{H}}_{\boldsymbol{d}}^{T}\left(\bar{\boldsymbol{g}} - \bar{\boldsymbol{H}}_{\boldsymbol{d}}\boldsymbol{f}\right) &= 0. \end{cases} \tag{6.20}$$

It is now immediate to apply the same derivation as in [27] and demonstrate that the left hand side of the first equation in both system (6.20) and system (6.19) are identical. Since the right hand sides are also identical, this implies that the first equations have the same solutions. The second equations in (6.20) and (6.19) are instead identical by construction. ∎

Notice that the new formulation requires the definition of a square and symmetric matrix $\boldsymbol{H}_{\boldsymbol{d}}^{\perp}$. This matrix depends on the parameter α and the prior matrix $\boldsymbol{\Sigma}$, both of which are unknown. However, for the purpose of estimating the unknown blur scale map $\boldsymbol{d}$, it is possible to bypass the estimation of α and $\boldsymbol{\Sigma}$ by learning directly the matrix $\boldsymbol{H}_{\boldsymbol{d}}^{\perp}$ from data.

6.2.1 Learning Procedure and Blur Scale Identification

The complexity of solving equation (6.6) is broken down by using local blur uniformity, *i.e.* by assuming that blur is constant within a small region of pixels. Then, we further simplify the problem by considering only a finite set of L blur sizes $d_1, \ldots, d_L$. In practice, both assumptions work well. The local blur uniformity holds reasonably well except at occluding boundaries, which form a small subset of the image domain. At occluding boundaries the solution tends to favour small blur estimates. It has been seen experimentally that the discretization is not a limiting factor in this method. The number of blur sizes L can be set to a value that matches the level of accuracy of the method without reaching a prohibitive computational load.

Now, by combining the assumptions, equation (6.6) can be written for one pixel $\mathbf{p}$ as

$$\boldsymbol{d}^*(\mathbf{p}) = \underset{\boldsymbol{d}(\mathbf{p})}{\operatorname{argmin}} \|\boldsymbol{H}_{\boldsymbol{d}}^{\perp}(\mathbf{p})\boldsymbol{g}\|_2^2 + \beta\|\nabla \boldsymbol{d}(\mathbf{p})\|_1 \tag{6.21}$$

can be approximated by

$$\boldsymbol{d}^*(\mathbf{p}) = \underset{\boldsymbol{d}(\mathbf{p})}{\operatorname{argmin}} \|\boldsymbol{H}_{\boldsymbol{d}(\mathbf{p})}^{\perp}\boldsymbol{g}_{\mathbf{p}}\|_2^2 \tag{6.22}$$

where $\boldsymbol{g}_{\mathbf{p}}$ is a column vector of δ^2 pixels extracted from a $\delta \times \delta$ patch centered at the pixel $\mathbf{p}$ of $\boldsymbol{g}$. It is found experimentally that the size δ of the patch should not be smaller than the maximum scale of the coded blur in the captured image $\boldsymbol{g}$. $\boldsymbol{H}_{\boldsymbol{d}(\mathbf{p})}^{\perp}$ is a $\delta^2 \times \delta^2$ matrix that depends on the blur size $\boldsymbol{d}(\mathbf{p}) \in \{d_1, \ldots, d_L\}$. It is assumed that $\boldsymbol{H}_{\boldsymbol{d}}^{\perp}(\mathbf{p}, \mathbf{y}) \simeq 0$ for $\mathbf{y}$ such that $\|\mathbf{y} - \mathbf{p}\|_1 > \delta/2$. Notice that the term $\beta\|\nabla \boldsymbol{d}\|_1$ drops because of the local blur uniformity assumption. The next step is to explicitly compute $\boldsymbol{H}_{\boldsymbol{d}(\mathbf{p})}^{\perp}$.

Learning procedure. Since the blur size $\boldsymbol{d}(\mathbf{p})$ is one of L values, there is only the need to compute $\boldsymbol{H}_{d_1}^{\perp}, \ldots, \boldsymbol{H}_{d_L}^{\perp}$ matrices. As each $\boldsymbol{H}_{d_i}^{\perp}$ depends on α and the local $\boldsymbol{\Sigma}$, one can learn each $\boldsymbol{H}_{d_i}^{\perp}$ directly from data. Suppose that one is given a set of T column

vectors $\boldsymbol{g}_{\mathbf{p}_1}, \ldots, \boldsymbol{g}_{\mathbf{p}_T}$ extracted from blurry images of a plane parallel to the camera image plane. The column vectors will all share the same blur scale d_i. Hence, the cost functional in equation (6.22) can be rewritten for all $\mathbf{p}$ as

$$\|\boldsymbol{H}_{d_i}^{\perp}\boldsymbol{G}_i\|_2^2 \tag{6.23}$$

where $\boldsymbol{G}_i \doteq [\boldsymbol{g}_{\mathbf{p}_1} \ \cdots \ \boldsymbol{g}_{\mathbf{p}_T}]$. By definition of $\boldsymbol{G}_i$, $\|\boldsymbol{H}_{d_i}^{\perp}\boldsymbol{G}_i\|_2^2 = 0$. Hence, we find that $\boldsymbol{H}_{d_i}^{\perp}$ can be computed via the singular value decomposition of $\boldsymbol{G}_i = U_i S_i V_i^T$. If $U_i = [\boldsymbol{Q}_{d_i} \ \boldsymbol{U}_{d_i}]$, where $\boldsymbol{U}_{d_i}$ corresponds to the singular values of S_i that are zero (or negligible), then

$$\boldsymbol{H}_{d_i}^{\perp} = \boldsymbol{U}_{d_i}\boldsymbol{U}_{d_i}^T. \tag{6.24}$$

The procedure is then repeated for each blur scale d_i with $i = 1, \ldots, L$.

The estimated matrices $\boldsymbol{H}_{d_1}^{\perp}, \ldots, \boldsymbol{H}_{d_L}^{\perp}$ can now be used on a new image $\boldsymbol{g}$ and optimize with respect to $\boldsymbol{d}$:

$$\boldsymbol{d}^* = \underset{\boldsymbol{d}}{\operatorname{argmin}} \sum_{\mathbf{p}} \|\boldsymbol{H}_{\boldsymbol{d}(\mathbf{p})}^{\perp}\boldsymbol{g}_{\mathbf{p}}\|_2^2 + \beta\|\nabla\boldsymbol{d}(\mathbf{p})\|_1. \tag{6.25}$$

The first term represents unitary terms, *i.e.*, terms that are defined on single pixels; the second term represents binary terms, *i.e.*, terms that are defined on pairs of pixels. The minimization problem (6.25) can then be solved efficiently via graph cuts [46]. The blur scale identification procedure is summarized in Algorithm 1.

Notice that the procedure above can be applied to other surfaces as well, so that instead of a collection of parallel planes one can consider, for example, a collection of quadratic surfaces. Also, there are no restrictions on the size of a patch. In particular, the same procedure can be applied to a patch of the size of the input image. In the experiments for depth estimation, however, only small patches and parallel planes as local surfaces are considered.

Input: A single coded image $\boldsymbol{g}$ and a collection of coded images of L planar scenes.
Output: The blur scale map $\boldsymbol{d}$ of the scene.
Preprocessing (offline)
Pick an image patch size larger than twice the maximum blur scale;
for $i = 1, \ldots, L$ **do**
  Compute the singular value decomposition $U_i S_i V_i^T$ of a collection of image patches coded with blur scale d_i ;
  Calculate the subspace $\boldsymbol{U}_{d_i}$ as the columns of U_i corresponding to singular values of S_i;
  Calculate the projection matrix $\boldsymbol{H}_{d_i}^{\perp} = \boldsymbol{U}_{d_i}\boldsymbol{U}_{d_i}^T$;
end
Blur identification (online)
Solve $\boldsymbol{d}^* = \arg\min_{\boldsymbol{d}\in\{d_1,\cdots,d_L\}} \sum_{\mathbf{p}} \|\boldsymbol{H}_{\boldsymbol{d}}^{\perp}\boldsymbol{g}_{\mathbf{p}}\|_2^2 + \beta\|\nabla \boldsymbol{d}(\mathbf{p})\|_1$.

Algorithm 1: Blur scale identification from a single coded image via the subspaces based method.

6.2.2 Image Deblurring

The previous section described the construction of a procedure to compute the blur scale $\boldsymbol{d}^*$ at each pixel. This section assumes that $\boldsymbol{d}^*$ is given and devise a procedure to compute the image $\boldsymbol{f}$. In principle, one could use the closed-form solution

$$\boldsymbol{f} = \left(\alpha \boldsymbol{\Sigma}^T\boldsymbol{\Sigma} + \boldsymbol{H}_{\boldsymbol{d}^*}^T\boldsymbol{H}_{\boldsymbol{d}^*}\right)^{-1}\boldsymbol{H}_{\boldsymbol{d}^*}^T\boldsymbol{g}. \tag{6.26}$$

However, notice that computing this equation entails solving a large matrix inversion, which is not practical for moderate image dimensions. A simpler approach is to solve the least squares problem (6.5) in $\boldsymbol{f}$ via an iterative method. Therefore, it is possible to consider solving the problem

$$\boldsymbol{f}^* = \underset{\boldsymbol{f}}{\operatorname{argmin}} \|\boldsymbol{g} - \boldsymbol{H}_{\boldsymbol{d}^*}\boldsymbol{f}\|_2^2 + \alpha\|\boldsymbol{\Sigma}\boldsymbol{f}\|_2^2 \tag{6.27}$$

by using a least-squares conjugate gradient descent algorithm in $\boldsymbol{f}$ [75]. The main component for the iteration in $\boldsymbol{f}$ is the gradient $\nabla E_{\boldsymbol{f}}$ of the cost (6.27) with respect to

$\boldsymbol{f}$

$$\nabla E_{\boldsymbol{f}} = \left(\alpha \boldsymbol{\Sigma}^T \boldsymbol{\Sigma} + \boldsymbol{H}_{\boldsymbol{d}^*}^T \boldsymbol{H}_{\boldsymbol{d}^*}\right) \boldsymbol{f} - \boldsymbol{H}_{\boldsymbol{d}^*}^T \boldsymbol{g}. \tag{6.28}$$

The descent algorithm iterates until $\nabla E_{\boldsymbol{f}} \simeq \boldsymbol{0}$. Because of the convexity of the cost functional with respect to $\boldsymbol{f}$, the solution is also a global minimum.

To compute $\boldsymbol{\Sigma}$ one can use a database of sharp images $\boldsymbol{F} = [\boldsymbol{f}_1 \cdots \boldsymbol{f}_T]$ (where $\{\boldsymbol{f}_i\}_{i=1,\ldots,T}$ are sharp images rearranged as column vectors), and compute the singular value decomposition $\boldsymbol{F} = U_F \Sigma_F V_F^T$. Then, one has to partition $U_F = [U_{F,1}\ U_{F,2}]$ such that $U_{F,2}$ corresponds to the smallest singular values of Σ_F. The high-order prior is defined as $\boldsymbol{\Sigma} \doteq U_{F,2} U_{F,2}^T$, so that $\boldsymbol{\Sigma} \boldsymbol{f}_i \approx 0$. The regularization parameter α is instead manually tuned.

6.3 Experiments

This section will demonstrate the effectiveness of the presented approach on both synthetic and real data. The algorithm performs better than previous methods on different coded apertures and different datasets. It is also shown that the masks proposed in the literature do not always yield the best performance.

6.3.1 Performance Comparison

Before proceeding with tests on real images, some extensive simulations are performed to compare accuracy and robustness of the algorithm proposed here with four competing methods including the current state-of-the-art approach. The methods are all based on the hypothesis plane deconvolution used by [50] as explained in the Introduction. The main difference among the competing methods is that the deconvolution step is performed either using the Lucy-Richardson method [89], or regularized filtering (*i.e.*, with image gradient smoothness), or Wiener filtering [5], or Levin's procedure [50]. All the eight masks shown in Figure 4.4 are tested. All the patterns have been proposed and used by other researchers [98, 50, 34, 39, 106, 64]. For each mask and a given blur scale map $\boldsymbol{d}$, a coded image is simulated by using equation (6.1), where $\boldsymbol{f}$

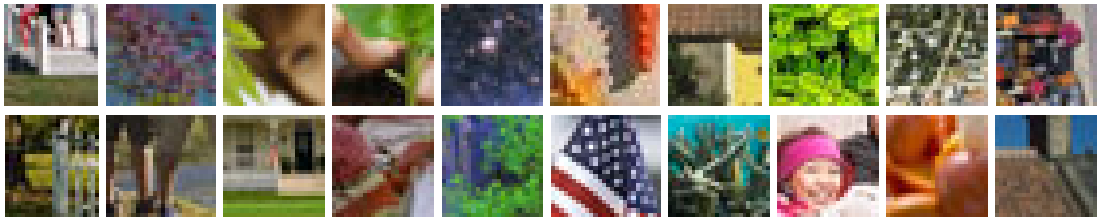


image noise level σ= 0

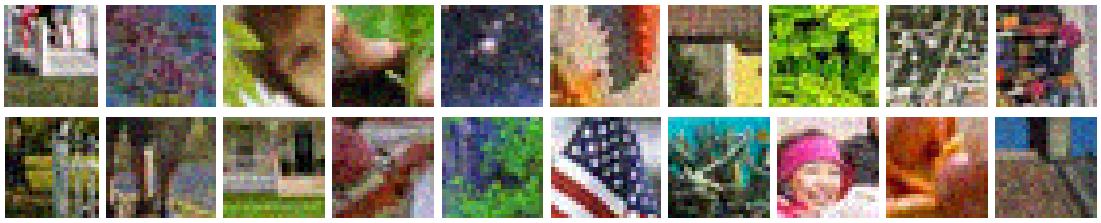


image noise level σ= 0.002

Figure 6.1: Patches of real texture. Some of the patches extracted from real images that have been used in our tests. The same patches are shown with no noise (top part) and when a Gaussian noise is added to them (bottom part).

is an image of $4,875 \times 125$ pixels with either random texture or a set of patches from natural images (examples of these patches are shown in Figure 6.1). Then, a blur scale map estimate $\hat{\boldsymbol{d}}$ is obtained for each algorithm and its discrepancy with the ground-truth is computed. The ground-truth blur scale map $\boldsymbol{d}$, used in the experiments, is shown in pseudo-colors at the top-left of both Figure 6.2 and Figure 6.3 and it represents a stair composed of 39 steps at different distances (and thus different blur scales) from the camera. It is assumed that the focal plane is set to be between the camera and the first object of interest in the scene. With this setting, the bottom part of the blur scale map (small blur sizes) corresponds to points close to the camera, and the top part (large blur sizes) to points far from the camera. Each step of the stair is a square of 125×125 pixels, but is has been squeezed along the vertical axis in the the actual illustration, to fit in the paper. The size of the blur ranges from 7 to 30 pixels. Notice that in measuring the errors all pixels are considered, including those at the blur scale discontinuities, given by the difference of blur scale between neighboring steps. Figure 6.2 reports, for each aperture mask in Figure 4.4, the results of the proposed method (right) together with the results obtained by the current state-of-the-art

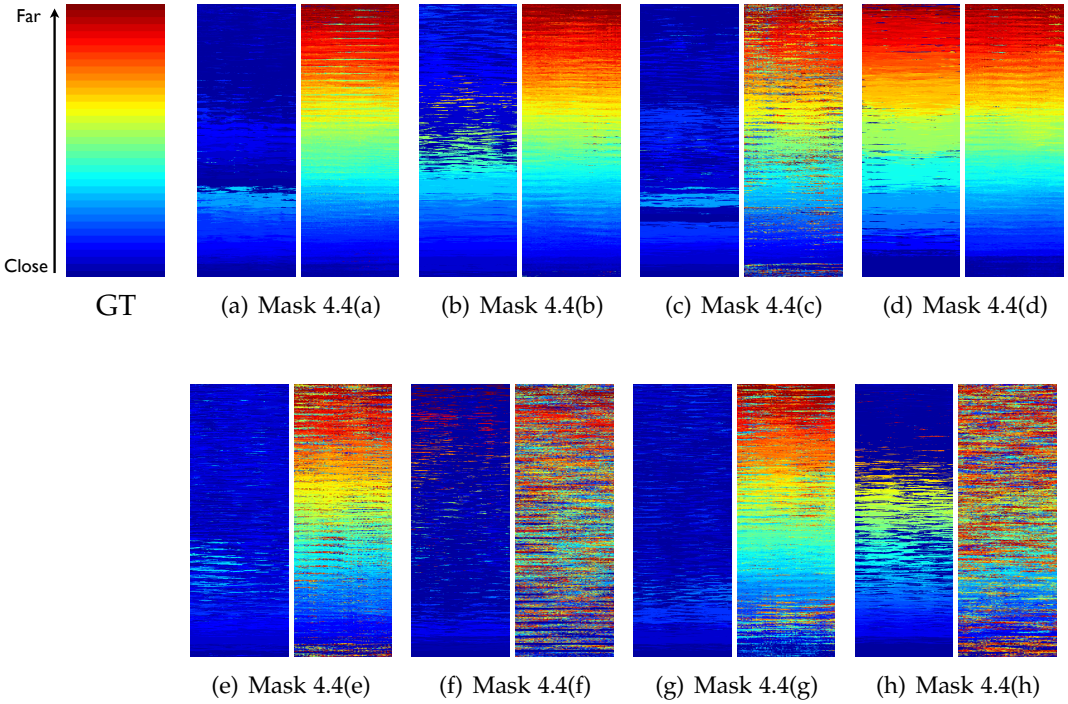


Figure 6.2: Blur scale estimation - random texture. GT: Ground-truth blur scale map. (a-h) Estimated blur scale maps for all the eight masks we consider in the paper. For each mask, the figure reports the blur scale map estimated with both Levin *et al.*'s method (left) and our method (right).

algorithm (left) on random texture. The same procedure, but with texture from natural images, is reported in Figure 6.3. For the three best performing aperture masks (mask 4.4(a), mask 4.4(b), and mask 4.4(d)), the results are reported with the same graphical layout in Figure 6.4, in order to better appreciate the improvement of this method over previous ones, especially for large blur scales. Every plot shows, for each of the 39 steps, the mean and 3 times the standard deviation of the estimated blur scale values (ordinate axis) against the true blur scale level (abscissa axis). The ideal estimate is the diagonal line where each estimated level corresponds to the correct true blur scale level. If there is no bias in the estimation of the blur scale map, the ideal estimate should lie between 3 times the standard deviation about the mean with probability close to 1. This method performs consistently well with all the masks and at different blur scale levels. In particular, the best performances are observed for the patterns

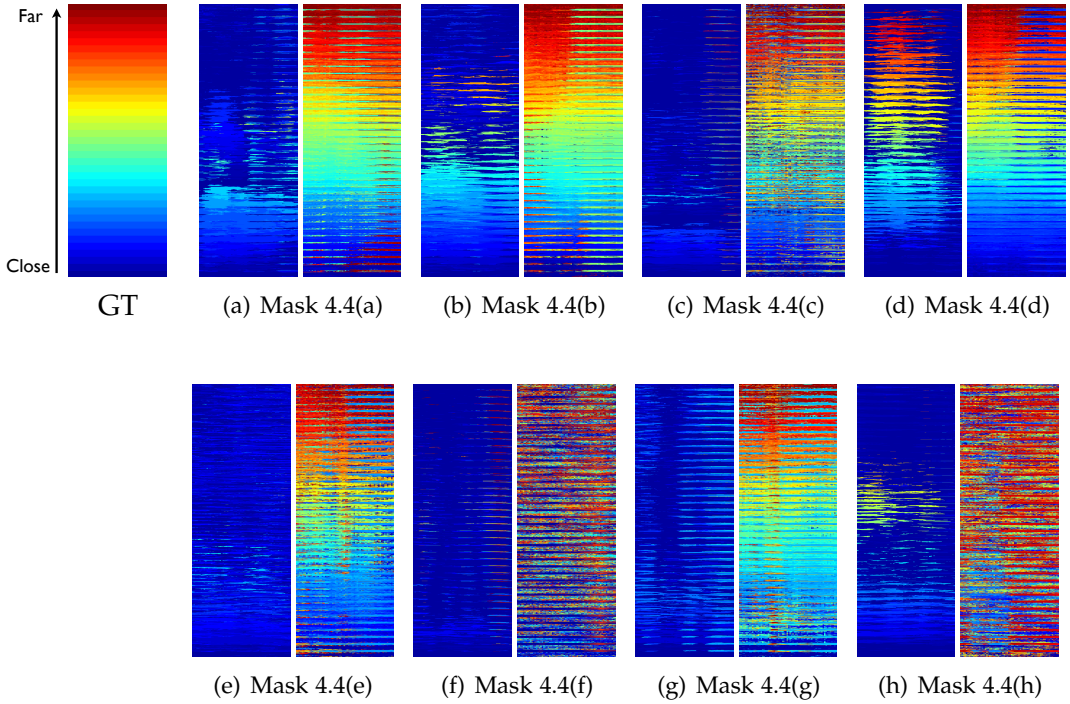
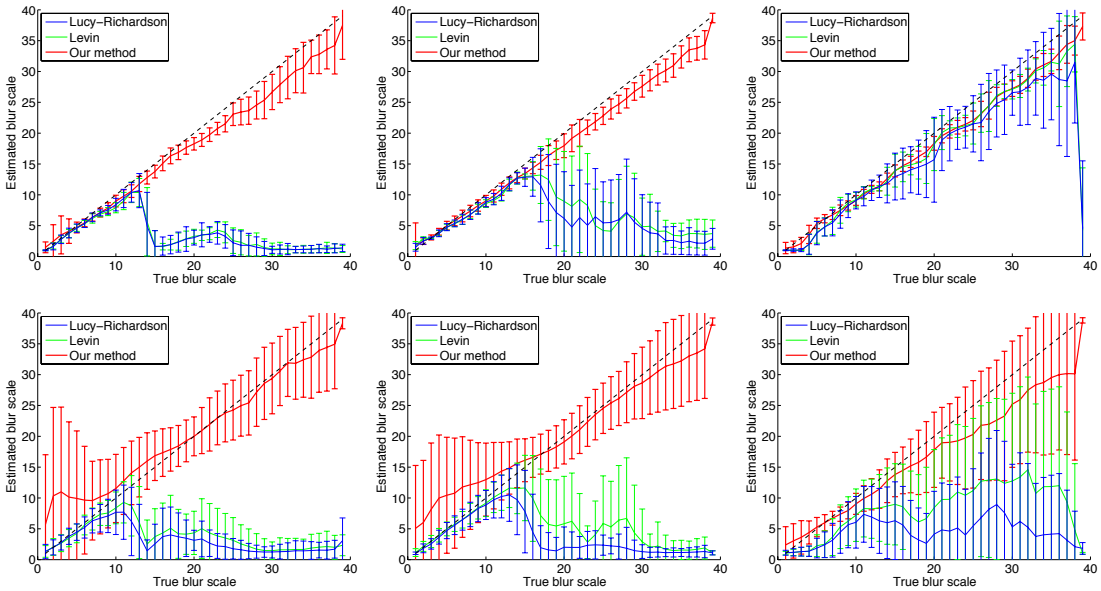


Figure 6.3: Blur scale estimation - real texture. GT: Ground-truth blur scale map. (a-h) Estimated blur scale maps for all the eight masks we consider in the paper. For each mask, the figure reports the blur scale map estimated with both Levin *et al.*'s method (left) and our method (right).

of the aperture mask 4.4(b) (Figure 6.4(b)) and the mask 4.4(d) (Figure 6.4(c)), while the performance of competing methods rapidly degenerates with increasing pattern scales. This demonstrates that this method has potential for restoring objects at a wider range of blur scales and with higher accuracy than previous algorithms.

A quantitative comparison about depth estimation among all the methods and masks is given in Tables 6.1 and 6.3 for random texture, and in Tables 6.2 and 6.4 for real texture. Each table reports the average error of the blur scale estimate, measured as $||\boldsymbol{d} - \boldsymbol{d}^*||_1$, where $\boldsymbol{d}$ and $\boldsymbol{d}^*$ are the ground-truth and the estimated blur scale map respectively.

The comparison about the deblurring procedure, instead, is given in Tables 6.5 and 6.7 for random texture, and in Tables 6.6 and 6.8 for real texture. The error on the reconstructed sharp image $\boldsymbol{f}^*$ is measured as $\sqrt{||\boldsymbol{f} - \boldsymbol{f}^*||_2^2 + ||\nabla \boldsymbol{f} - \nabla \boldsymbol{f}^*||_2^2}$, where



(a) Mask 4.4(a) (b) Mask 4.4(b) (c) Mask 4.4(d)

Figure 6.4: Blur scale estimation comparison for the 3 best performing methods, using both random (top) and real (bottom) texture. Each graph reports the performance of the algorithms with (a) masks 4.4(a), (b) masks 4.4(b), and (c) mask 4.4(d). Both mean and standard deviation (in the graphs, we show three times the computed standard deviation) of the estimated blur scale are shown in an error bar with the algorithms performances (solid lines) over the ideal characteristic curve (diagonal dashed line) for 39 blur sizes. Notice how the performance dramatically changes based on the nature of texture (top row vs bottom row). Moreover, in the case of real images the standard deviation of the estimates obtained with our method are more uniform for mask 4.4(b) than for mask 4.4(d). In the case of mask 4.4(d) the performance is reasonably accurate only with small blur scales.

$\boldsymbol{f}$ is the ground-truth image. The gradient term is added to improve sensitivity to artifacts in the reconstruction. Several levels of noise have been considered in the performance comparison: $\sigma = 0$ (Tables 6.1, 6.2, 6.5, and 6.6), $\sigma = 0.001$, $\sigma = 0.002$, and $\sigma = 0.005$ (Tables 6.3, 6.4, 6.7, and 6.8). The noise level is however adjusted to accommodate the difference in overall incoming light between the masks, i.e., if the

Methods	**Masks** - (image noise level $\sigma = 0$)							
	a	b	c	d	e	f	g	h
Lucy-Richardson	16.8	14.4	17.2	2.9	17.0	18.1	17.8	15.4
Regularized filtering	18.4	17.2	18.6	6.8	16.7	12.3	18.8	13.4
Wiener filtering	8.8	13.8	14.4	16.6	16.3	15.3	14.1	15.3
Levin *et al.*	16.7	13.7	16.7	1.4	16.6	16.8	17.6	13.3
Our method	**1.2**	**0.9**	**3.7**	**0.9**	**4.2**	**10.3**	**3.8**	**9.6**

Table 6.1: Blur estimation with random texture (without noise). Performance (mean error) of 5 algorithms in blur scale estimation for the apertures in Figure 4.4, assuming there is not noise.

Methods	**Masks** - (image noise level $\sigma = 0$)							
	a	b	c	d	e	f	g	h
Lucy-Richardson	17.0	16.4	18.4	15.6	17.9	18.5	18.0	18.3
Regularized filtering	18.5	16.8	18.2	8.6	16.8	11.4	17.9	15.4
Wiener filtering	17.1	16.4	18.2	14.4	17.0	18.0	17.5	17.6
Levin *et al.*	16.3	14.8	17.9	9.9	17.0	18.2	17.6	17.0
Our method	**3.3**	**3.3**	**6.8**	**3.3**	**6.1**	**12.6**	**5.9**	**11.7**

Table 6.2: Blur estimation with real texture (without noise). Performance (mean error) of 5 algorithms in blur scale estimation for the apertures in Figure 4.4, assuming there is not noise.

mask i has an incoming light of l_i[1], the noise level for that mask is given by:

$$\sigma_i = \frac{1}{l_i} * \sigma. \tag{6.29}$$

Thus, masks such as 4.4(f), 4.4(g) and 4.4(h) are subject to lower noise levels than masks such as 4.4(a) and 4.4(b). The proposed method produces more consistent and accurate blur scale maps than previous methods for both random texture and natural images, and across the eight masks that it has been tested with.

With the increasing of the noise of the input image, less layers (or blur scales) can be distinguished in the blur map, especially for big amounts of blur. When the noise level $\sigma > 0.005$, the estimation is very poor for all the five methods. The worst estimation happens when the reconstructed blur map is just one single layer: this yields to a maximum error that is half of the number of blur scales that are considered

[1]The value of l_i represents the quantity of lens aperture that is open: when the lens aperture is totally open, $l_i = 1$; instead, when the mask completely blocks the light, $l_i = 0$.

Methods	**Masks** - (image noise level $\sigma = 0.001$)							
	a	b	c	d	e	f	g	h
Lucy-Richardson	18.5	17.1	18.2	11.7	16.6	16.2	18.3	17.3
Regularized filtering	19.0	17.5	19.0	14.3	16.8	18.3	18.9	15.6
Wiener filtering	15.7	16.7	16.8	17.5	17.2	17.6	16.8	17.0
Levin *et al.*	18.4	16.3	18.1	11.0	16.7	17.3	18.3	17.5
Our method	**9.6**	**8.7**	**12.7**	**10.1**	**12.5**	**13.2**	**12.9**	**13.9**
Methods	**Masks** - (image noise level $\sigma = 0.002$)							
	a	b	c	d	e	f	g	h
Lucy-Richardson	18.5	17.1	18.2	12.1	16.6	16.3	18.3	17.3
Regularized filtering	18.9	17.4	18.8	12.7	16.7	16.9	18.9	16.9
Wiener filtering	15.5	16.4	16.7	17.3	17.1	17.5	16.8	17.0
Levin *et al.*	18.5	16.9	18.0	12.1	16.7	17.6	18.4	17.7
Our method	**11.3**	**11.1**	**13.2**	**11.3**	**12.6**	**13.5**	**12.8**	**14.0**
Methods	**Masks** - (image noise level $\sigma = 0.005$)							
	a	b	c	d	e	f	g	h
Lucy-Richardson	18.4	17.0	18.2	12.6	16.5	16.6	18.4	17.3
Regularized filtering	18.9	17.4	18.8	13.1	16.6	17.1	18.8	16.9
Wiener filtering	15.4	16.2	16.5	17.3	17.2	17.3	16.7	17.0
Levin *et al.*	18.5	16.9	18.0	12.5	16.7	17.7	18.4	17.7
Our method	**12.8**	**12.6**	**13.4**	**12.0**	**12.8**	**13.5**	**13.5**	**14.0**

Table 6.3: Blur estimation with random texture (with noise). Performance (mean error) of 5 algorithms in blur scale estimation for the aperture masks in Figure 4.4, under different levels of noise.

Methods	**Masks** - (image noise level $\sigma = 0.001$)							
	a	b	c	d	e	f	g	h
Lucy-Richardson	18.5	17.2	18.3	13.7	16.8	17.8	18.4	18.1
Regularized filtering	19.0	17.5	19.0	14.0	16.8	17.6	19.0	15.6
Wiener filtering	13.8	14.5	14.1	14.6	15.2	14.4	14.8	14.5
Levin *et al.*	18.4	16.8	18.1	10.6	16.7	17.0	18.2	17.8
Our method	**8.7**	**7.8**	**11.8**	**7.7**	**11.9**	**13.5**	**11.5**	**13.8**
Methods	**Masks** - (image noise level $\sigma = 0.002$)							
	a	b	c	d	e	f	g	h
Lucy-Richardson	18.5	17.2	18.3	13.2	16.7	17.5	18.4	17.9
Regularized filtering	19.0	17.5	19.0	14.1	16.8	18.1	19.0	15.7
Wiener filtering	14.7	15.8	15.2	15.8	16.0	15.1	15.0	15.7
Levin *et al.*	18.4	16.8	18.1	11.1	16.7	17.1	18.3	17.7
Our method	**10.6**	**9.5**	**12.1**	**9.0**	**12.3**	**13.5**	**12.1**	**14.1**
Methods	**Masks** - (image noise level $\sigma = 0.005$)							
	a	b	c	d	e	f	g	h
Lucy-Richardson	18.3	17.1	18.2	12.9	16.6	17.4	18.4	17.9
Regularized filtering	19.0	17.5	19.0	14.1	16.8	18.1	18.9	15.7
Wiener filtering	15.6	16.5	16.1	16.8	16.7	16.5	16.0	16.7
Levin *et al.*	18.5	16.9	18.1	11.3	16.7	17.4	18.4	17.7
Our method	**12.2**	**11.8**	**13.3**	**10.8**	**12.7**	**13.7**	**13.4**	**13.7**

Table 6.4: Blur estimation with real texture (with noise). Performance (mean error) of 5 algorithms in blur scale estimation for the aperture masks in Figure 4.4, under different levels of noise.

in the test (in our case the maximum error for blur estimation is ~ 20).

Methods	**Masks** - (image noise level $\sigma = 0$)							
	a	b	c	d	e	f	g	h
Lucy-Richardson	0.22	0.22	0.21	0.22	0.22	0.22	0.22	**0.21**
Regularized filtering	0.30	0.32	0.27	0.32	0.25	0.42	0.23	0.25
Wiener filtering	0.23	0.29	0.29	0.33	0.31	0.32	0.27	0.30
Levin *et al.*	0.22	0.21	0.22	**0.21**	**0.21**	**0.22**	0.22	**0.21**
Our method	**0.20**	**0.20**	**0.21**	**0.21**	**0.21**	**0.22**	**0.21**	0.22

Table 6.5: Image deblurring with random texture (without noise). Performance (mean error) of 5 algorithms in image deblurring for the apertures in Figure 4.4, assuming there is not noise.

Methods	**Masks** - (image noise level $\sigma = 0$)							
	a	b	c	d	e	f	g	h
Lucy-Richardson	0.22	0.20	0.22	0.18	0.20	0.20	0.20	**0.20**
Regularized filtering	0.51	0.49	0.52	1.08	0.28	0.67	0.28	0.40
Wiener filtering	0.25	0.22	0.26	0.21	0.21	0.24	0.23	0.21
Levin *et al.*	0.25	0.21	0.23	0.19	0.20	**0.21**	0.21	**0.20**
Our method	**0.18**	**0.16**	**0.21**	**0.16**	**0.17**	**0.21**	**0.19**	0.21

Table 6.6: Image deblurring with real texture (without noise). Performance (mean error) of 5 algorithms in image deblurring for the apertures in Figure 4.4, assuming there is not noise.

6.3.2 Results on Real Data

The proposed blur scale estimation algorithm is now applied to coded aperture images captured by inserting the selected mask into a Canon 50mm $f/1.4$ lens mounted on a Canon EOS-5D DSLR as described in [50, 106]. Based on the performance analysis from the previous section, the aperture masks 4.4(b) and 4.4(d) were chosen for our experiments. Each of the four holes in the first mask is $3.5mm$ large, which corresponds to the same overall section of a conventional (circular) aperture with diameter $7.9mm$ ($f/6.3$ in a $50mm$ lens). All indoor images have been captured by setting the shutter speed to $30ms$ (ISO 320-500) while outdoors the exposure has been set to $2ms$ or lower (ISO 100).

Firstly, one needs to collect (or synthesize) a sequence of L coded images, where L is the number of blur scale levels we want to distinguish. There are two techniques to acquire these coded images: (1) If the aim is just to estimate the depth map (or blur

Methods	**Masks** - (image noise level $\sigma = 0.001$)							
	a	b	c	d	e	f	g	h
Lucy-Richardson	0.39	0.36	0.27	0.28	0.35	0.29	0.26	0.27
Regularized filtering	0.88	0.96	0.61	1.03	0.93	0.61	0.61	0.91
Wiener filtering	0.35	0.37	0.36	0.39	0.38	0.38	0.35	0.38
Levin *et al.*	0.32	0.31	0.26	0.28	0.30	0.28	0.25	0.26
Our method	**0.20**	**0.21**	**0.22**	**0.22**	**0.21**	**0.23**	**0.21**	**0.23**
Methods	**Masks** - (image noise level $\sigma = 0.002$)							
	a	b	c	d	e	f	g	h
Lucy-Richardson	0.49	0.46	0.31	0.34	0.44	0.33	0.30	0.32
Regularized filtering	0.76	0.69	0.47	0.50	0.67	0.46	0.49	0.46
Wiener filtering	0.35	0.37	0.37	0.39	0.38	0.39	0.35	0.38
Levin *et al.*	0.39	0.38	0.29	0.34	0.37	0.31	0.28	0.29
Our method	**0.22**	**0.22**	**0.23**	**0.23**	**0.22**	**0.23**	**0.23**	**0.24**
Methods	**Masks** - (image noise level $\sigma = 0.005$) Image deblurring							
	a	b	c	d	e	f	g	h
Lucy-Richardson	0.66	0.62	0.41	0.47	0.61	0.40	0.40	0.43
Regularized filtering	1.17	1.04	0.69	0.75	1.03	0.59	0.73	0.68
Wiener filtering	0.35	0.37	0.37	0.39	0.38	0.39	0.35	0.38
Levin *et al.*	0.55	0.54	0.37	0.45	0.51	0.37	0.36	0.39
Our method	**0.25**	**0.25**	**0.26**	**0.25**	**0.25**	**0.26**	**0.26**	**0.27**

Table 6.7: Image deblurring with random texture (with noise). Performance (mean error) of 5 algorithms in image deblurring for the aperture masks in Figure 4.4, under different levels of noise.

Methods	**Masks** - (image noise level $\sigma = 0.001$)							
	a	b	c	d	e	f	g	h
Lucy-Richardson	0.38	0.35	0.26	0.24	0.32	0.23	0.24	0.25
Regularized filtering	0.96	1.05	0.66	1.39	0.94	0.68	0.64	1.02
Wiener filtering	0.21	0.23	0.22	0.22	0.23	0.21	0.21	0.23
Levin *et al.*	0.34	0.33	0.27	0.30	0.30	0.27	0.24	0.25
Our method	**0.21**	**0.18**	**0.22**	**0.17**	**0.19**	**0.20**	**0.20**	**0.20**
Methods	**Masks** - (image noise level $\sigma = 0.002$)							
	a	b	c	d	e	f	g	h
Lucy-Richardson	0.47	0.44	0.30	0.29	0.40	0.26	0.27	0.30
Regularized filtering	1.26	1.38	0.87	1.72	1.30	0.74	0.87	1.34
Wiener filtering	**0.23**	0.25	0.24	0.24	0.25	0.24	0.22	0.25
Levin *et al.*	0.41	0.40	0.30	0.37	0.37	0.30	0.28	0.29
Our method	0.24	**0.19**	**0.23**	**0.17**	**0.19**	**0.20**	**0.21**	**0.20**
Methods	**Masks** - (image noise level $\sigma = 0.005$)							
	a	b	c	d	e	f	g	h
Lucy-Richardson	0.61	0.58	0.39	0.40	0.55	0.34	0.37	0.40
Regularized filtering	1.89	2.07	1.31	2.38	2.03	0.88	1.31	2.02
Wiener filtering	**0.26**	0.27	0.26	0.27	0.27	0.26	0.24	0.26
Levin *et al.*	0.56	0.55	0.38	0.49	0.51	0.37	0.35	0.39
Our method	**0.26**	**0.22**	**0.24**	**0.19**	**0.21**	**0.22**	**0.22**	**0.25**

Table 6.8: Image deblurring with real texture (with noise). Performance (mean error) of 5 algorithms in image deblurring for the aperture masks in Figure 4.4, under different levels of noise.

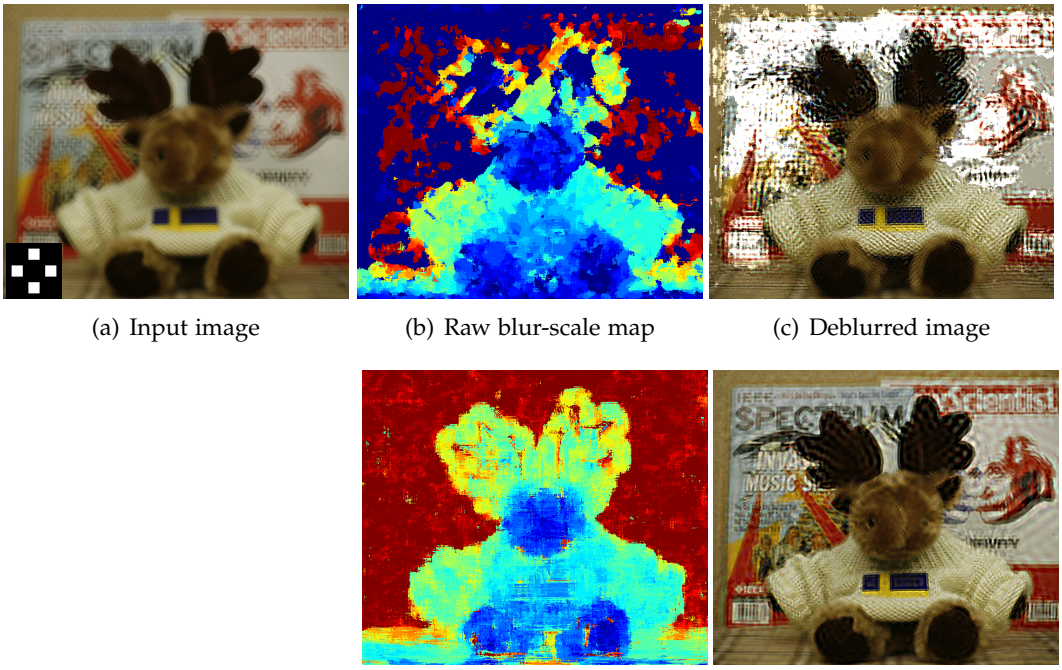
scale map), one can capture real coded images of a planar surface with sharp natural

(a) Conventional aperture (b) Ground-truth (pinhole camera)

Figure 6.5: Conventional aperture and pinhole camera. (a) Picture taken with the conventional camera without placing the mask on the lens. (b) Image captured by simulating a pinhole camera (f/22.0), which can be used as ground-truth for the image texture.

texture (e.g., a newspaper) at different blur scale levels. (2) If the goal is to reconstruct both depth map and all-in-focus image, one has to capture the PSF of the camera at each depth level, by projecting a grid of bright dots on a plane and using a long exposure; then, coded images are simulated by applying the measured PSFs on sharp natural images collected from the web. In the experiments presented here, the latter approach is used, since both blur scale map and all-in-focus image are estimated. The PSFs have been captured on a plane at 40 different depths between 60cm and 140cm from the camera. The focal plane of the camera was set at 150cm.

The first experiments demonstrate the advantage of the presented approach over Levin *et al.*'s method on a scene with blur sizes similar to the ones used in the performance test. The same dataset has been captured by using mask 4.4(b) (see Figure 6.6) and mask 4.4(d) (see Figure 6.7). The size of the blur, especially at the background, is very large; This can be appreciated in Figure 6.5(a), which shows the same scenario captured with the same camera setting, but without mask on the lens. For a fair comparison, no regularization or user intervention are used to the estimated blur scale maps. As already seen in the Section 6.3.1 (especially in Figure 6.4), Levin *et al.*'s



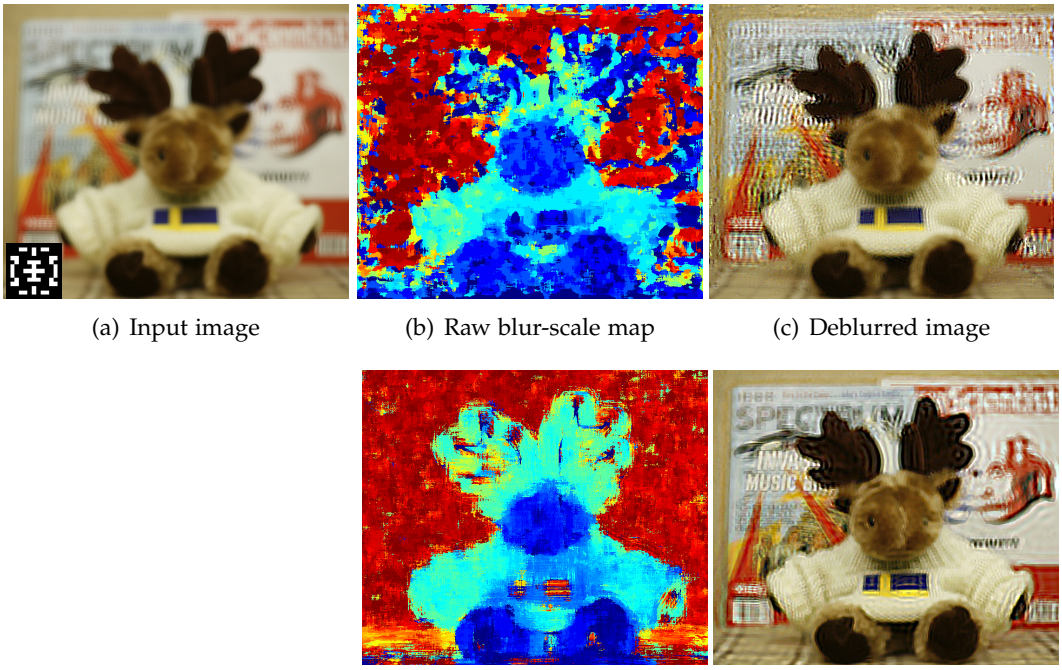
(a) Input image (b) Raw blur-scale map (c) Deblurred image

(d) Raw blur-scale map (e) Deblurred image

Figure 6.6: Comparison on real data - mask 4.4(b). (a) Input image captured by using mask 4.4(b). (b-c) Blur-scale map and all-in-focus image reconstructed with Levins *et al.*'s method [50]; (d-e) Results obtained from our method.

method yields an accurate blur scale estimate with mask 4.4(d) when the size of the blur is small, but it fails with large amounts of blur. The proposed approach overcomes this limitation and yields to a deblurred image that in both cases, Figure 6.6(e) and Figure 6.7(e), is closer to the ground-truth (Figure 6.5(b)). Notice also that this method gives an accurate reconstruction of the blur scale, even without using regularization ($\beta = 0$ in equation (6.25)). Some artefacts are still present in the reconstructed all-in-focus images. These are mainly due to the very large size of the blur and to the raw blur-scale map: When adding regularization to the blur-scale map ($\beta > 0$), the deblurring algorithm yields to better results, as one can see in the next examples.

In Figure 6.8 there is the same indoor scenario, but now the items are slightly closer to the focal plane of the camera; then the maximum amount of blur is reduced. Although the background is still very blurred in the coded image (Figure 6.8(a)), our



(a) Input image (b) Raw blur-scale map (c) Deblurred image

(d) Raw blur-scale map (e) Deblurred image

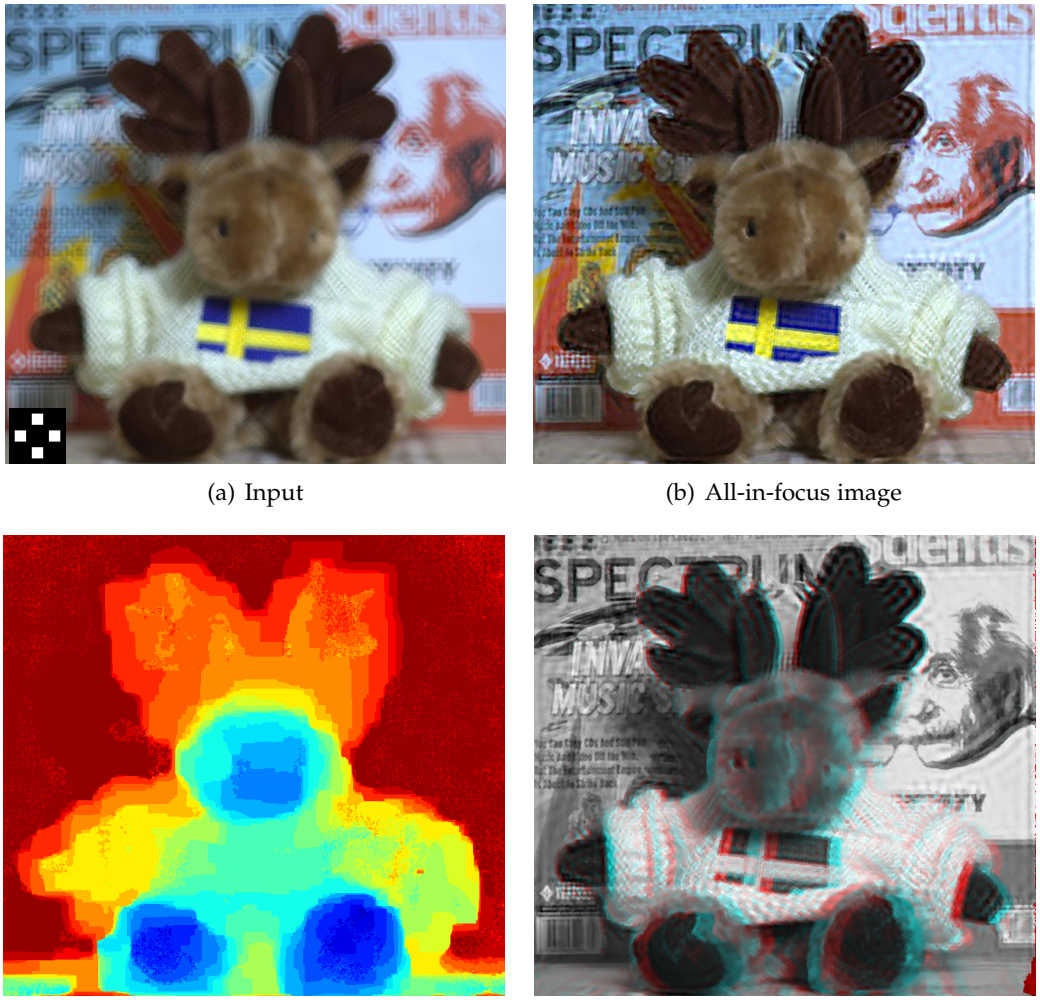
Figure 6.7: Comparison on real data - mask 4.4(d). (a) Input image captured by using mask 4.4(d). (b-c) Blur-scale map and all-in-focus image reconstructed with Levins *et al.*'s method [50]; (d-e) Results obtained from our method.

accurate blur-scale estimation yields to a deblurred image (Figure 6.8(b)), where the text of the magazine becomes readable. Since the reconstructed blur-scale map corresponds to the depth map (relative depth) of the scene, one can join it with the all-in-focus image to generate a *3D image*[1]. This image, when watched with red-cyan glasses, allows one to perceive the depth information extracted with our approach.

All the regularized blur-scale maps in this work are estimated from equation (6.25) by setting $\beta = 0.5$; the *raw* maps, instead, are obtained without regularization term ($\beta = 0$).

The proposed approach has been tested on different outdoor scenes: Figure 6.10 and Figure 6.9. The filters that are used in these scenarios have been learned within

[1]In this work, a *3D image* corresponds to an image captured with a stereo camera, where one lens has a *red* filter and the second lens has a *cyan* filter. When one watches this type of images with red-cyan glasses, each eye will see only one view: The shift between the two views gives the perception of depth.



(a) Input (b) All-in-focus image

(c) Blur-scale map (d) 3D image

Figure 6.8: Close-range indoor scene [exposure time: 1/30s]. (a) coded image captured with mask 4.4(b); (b) estimated all-in-focus image; (c) estimated blur-scale map; (d) 3D image (to be watched with red-cyan glasses).

150*cm* from the camera, but works even for a very large range of depths. Several challenges are present in these scenes, such as occlusions, shadows, and lack of texture. This method demonstrates robustness to all of them. Notice again that the raw blur-scale maps shown in Figure 6.10(c) and Figure 6.9(c) are already very close to the maps that include regularization (Figure 6.10(d) and Figure 6.9(d) respectively). For each dataset, a 3D image (Figure 6.9(e) and Figure 6.10(e)) has been generated by using

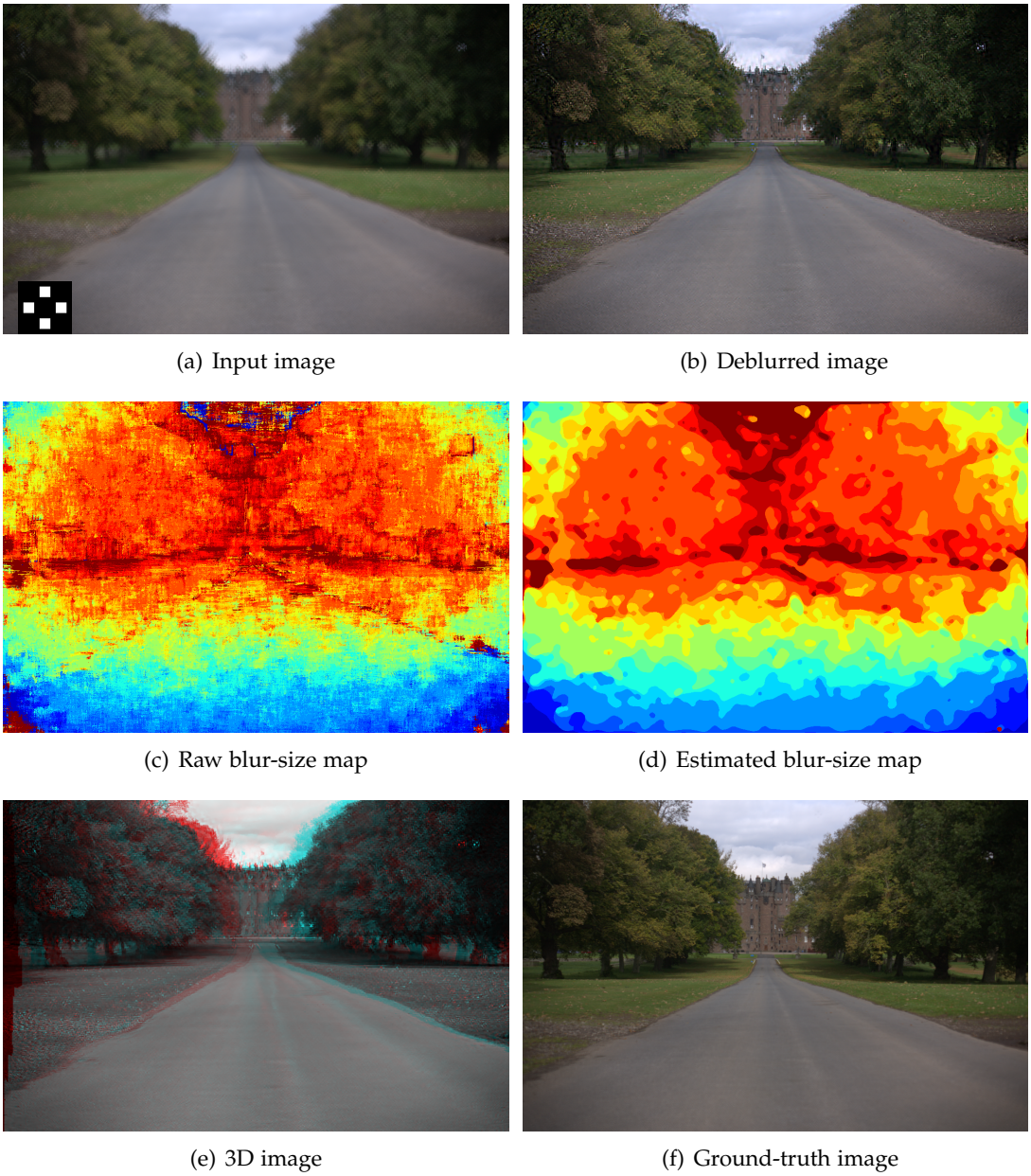
just the output of our method: the deblurred images (b) and the blur-scale maps (d). The ground-truth images have been taken by simulating a pinhole camera (f/22.0).

6.3.3 Computational Cost

The input images are downsampled 4 times from an original resolution of 12,8 megapixel ($4,368 \times 2,912$), and sub-pixel accuracy is used, in order to keep the algorithm efficient. It has been noticed from experiments on real data that the raw blur-scale map is already very close to the regularized map. This means that one can obtain a reasonable blur scale map very efficiently: When $\beta = 0$ the value of the blur scale at one pixel is independent of the other pixels and the calculations can be carried out in parallel. Since the algorithm takes about 5ms for processing 40 blur scale levels at each pixel, it is suitable for real-time applications. The algorithm was run on a QuadCore 2.8GHz with 16GB memory. The code has been written mainly in Matlab 7. The deblurring procedure, instead, takes about 100s to process the whole image for 40 blur scale levels.

6.4 Summary

This chapter presented a novel method to recover the all-in-focus image from a single blurred image captured with a coded aperture camera. The method is split in two steps: A subspace-based blur scale identification approach and an image deblurring algorithm based on conjugate gradient descent. The method is simple, general, and computationally efficient. A clear advantage of this method is that the training set can be obtained from real data, simply by capturing images of a plane at different distances from the camera. The proposed method has also been compared to existing algorithms in the literature and it was showed that it achieves state-of-the-art performance in blur scale identification and image deblurring with both synthetic and real data.

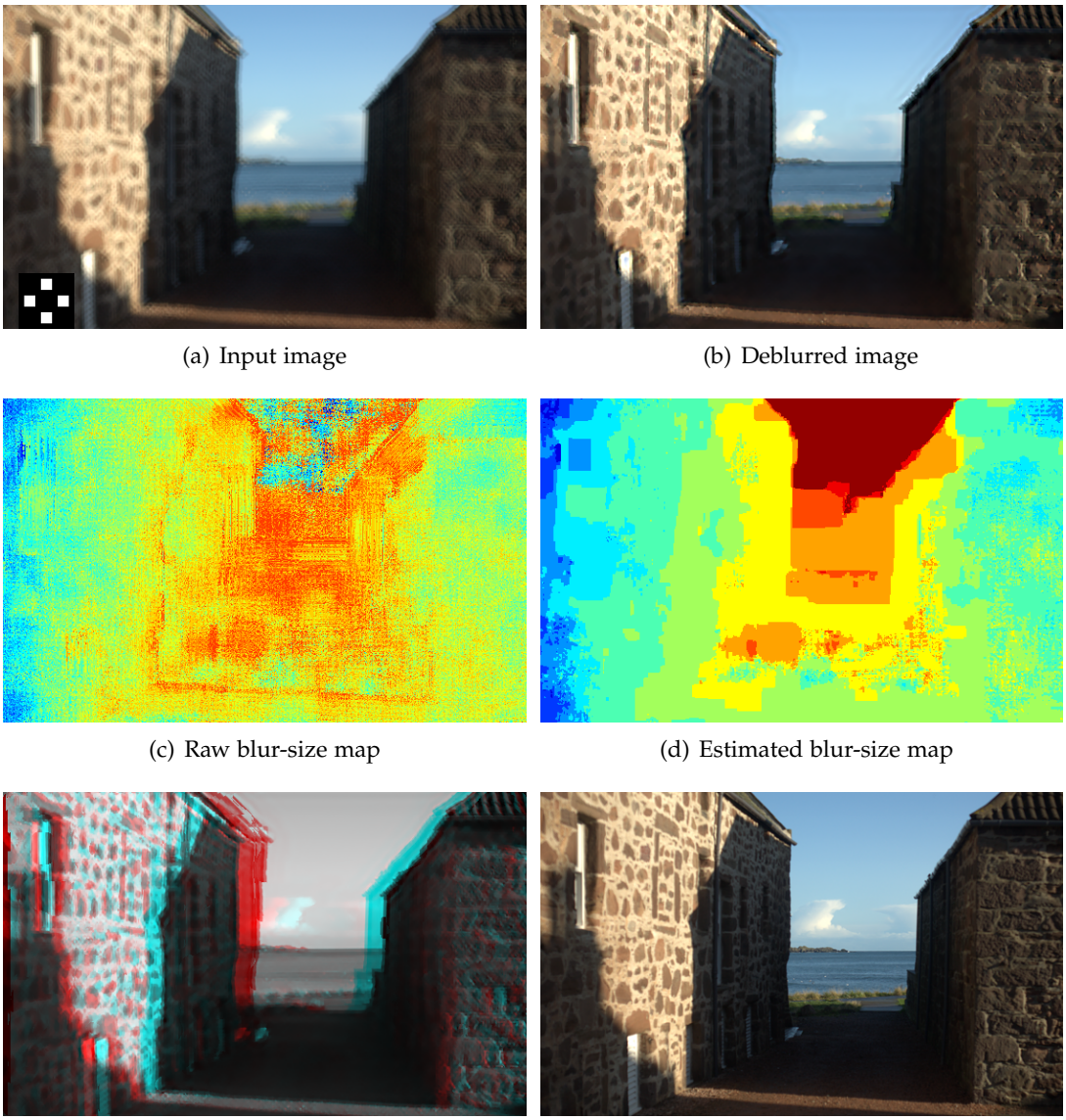


(a) Input image (b) Deblurred image

(c) Raw blur-size map (d) Estimated blur-size map

(e) 3D image (f) Ground-truth image

Figure 6.9: Long-range outdoor scene [exposure time: 1/200s]. (a) coded image captured with mask 4.4(b); (b) estimated all-in-focus image; (c) raw blur-scale map (without regularization); (d) regularized blur-scale map; (e) 3D image (to be watched with red-cyan glasses); (f) ground-truth image.



(a) Input image (b) Deblurred image

(c) Raw blur-size map (d) Estimated blur-size map

(e) 3D image (f) Ground-truth image

Figure 6.10: Mid-range outdoor scene [exposure time: 1/200s]. (a) coded image captured with mask 4.4(b); (b) estimated all-in-focus image; (c) raw blur-scale map (without regularization); (d) regularized blur-scale map; (e) 3D image (to be watched with red-cyan glasses); (f) ground-truth image.

This page has been left intentionally blank.

Chapter 7

Extension to Motion and Defocus Deblurring

If you can't make it good,
at least make it look good.
Bill Gates [1955 - present]

In the previous chapter, an all-in-focus image is obtained by removing the blur caused by defocus. However, if there are moving objects in the scene, they appear blurred in the image because of their motion. In this case, to recover the sharp texture, one has to identify and remove both defocus and motion blur. An example of this type of scenario is shown in Figure 7.1, where the captured image is affected by motion (the bus is moving) and defocus (the shops at the background are out-of-focus) deblurring.

This chapter introduces for the first time an efficient technique to identify and perform space-varying defocus and motion deblurring from a single image. The presented algorithm estimates both motion blur magnitude and direction as well as defocus blur scale at each pixel. It is also shown that, for the same overall incoming light, a coded aperture leads to better motion and defocus deblurring than a (compact) conventional circular aperture (Section 7.4). Finally, in Section 7.6 the method is tested

Figure 7.1: Challenging scene. Example of a blurred image captured with a conventional camera, where the degradation is due to both defocus (background) and motion (bus).

with both synthetic and real images.

7.1 Related Work

Motion and defocus deblurring from a single image when the scene is approximately a fronto-parallel plane has been long known in the field of signal processing as blind deconvolution [56, 65]. Recently, it has received renewed attention due to progress achieved by using natural image priors [42, 49, 86, 30]. For this choice of priors, currently [102] achieves the best results and can deal with very large (although uniform) blurs. Other recent methods deal with non-uniform motion-blur, but they assume that the scene is rigid and the motion is due to the camera shake [101]. An analysis of blind deconvolution algorithms in [52] finds that recovering blur first and then performing deblurring is a key ingredient. It also shows that the shift invariance assumption in all existing algorithms is often violated in real imagery. Our two-step approach for space-varying deblurring is somewhat inspired by these conclusions.

Alternative approaches to motion-deblurring are shown in [51], where a prototype camera moves with a parabolic motion during the exposure, and in [98, 3] where exposure is coded to facilitate the inversion of the motion-blur kernel. These techniques, however, have not been tested yet on images affected by space-varying defocus. Furthermore, as coding the exposure results in limiting the amount of incoming light, a

longer exposure is needed.

7.2 Motion and Defocus Deblurring

When imaging moving objects, the image undergoes a degradation that is made of both defocus, which depends on the aperture and the location $\boldsymbol{d}$ of the object in space, and motion blur, which depends on the object motion $\boldsymbol{m}$. Since objects in the scene may be placed at different locations and/or moving with different motions, the degradation (or blur) may be different at each pixel in the image. Hence, we use the general linear model, already studied in Section 3.3.2 and used in the previous chapter, to express a blurred image $\boldsymbol{g}$:

$$\boldsymbol{g} = \boldsymbol{H}_{\boldsymbol{d},\boldsymbol{m}}\boldsymbol{f} \ , \tag{7.1}$$

with the matrix $\boldsymbol{H}_{\boldsymbol{d},\boldsymbol{m}} = [\boldsymbol{h}_1 \ \boldsymbol{h}_2 \ \dots \ \boldsymbol{h}_M] \in \mathbb{R}^{M\times M}$, where M is the number of pixels of the image. Each blurring kernel $\boldsymbol{h}_i$ can be rearranged as a 2D matrix $\boldsymbol{h}_i^{\square}$, which can be thought as the result of a convolution between two simpler kernels

$$\boldsymbol{h}_i^{\square} = \boldsymbol{h}_{d_i}^{\square} * \boldsymbol{h}_{m_i}^{\square} \tag{7.2}$$

where $\boldsymbol{h}_{d_i}^{\square}$ contains only the degradation due to defocus and $\boldsymbol{h}_{m_i}^{\square}$ corresponds to the motion blur.

The problem of deblurring a single image can be posed as

$$\tilde{\boldsymbol{f}}, \tilde{\boldsymbol{d}}, \tilde{\boldsymbol{m}} = \underset{\boldsymbol{f},\boldsymbol{d},\boldsymbol{m}}{\operatorname{argmin}} \left[||\boldsymbol{g} - \hat{\boldsymbol{g}}||^2 + E_{reg}(\boldsymbol{f}, \boldsymbol{d}, \boldsymbol{m}) \right] \tag{7.3}$$

where we require the simulated blurred image $\hat{\boldsymbol{g}} = \boldsymbol{H}_{\boldsymbol{d},\boldsymbol{m}}\boldsymbol{f}$ to match in a least square sense the measured image $\boldsymbol{g}$, and we impose in the regularization term $E_{reg}(\boldsymbol{f}, \boldsymbol{d}, \boldsymbol{m})$ that all unknowns be piecewise constant. This minimization problem is a formidable challenge as we are given a single image $\boldsymbol{g}$ and we are looking for a 4-fold increase in number of parameters. Hence, to reduce the complexity of the problem we quantize the space of the scale and motion parameters so that only a finite set of possible values

is allowed. Moreover, we break down the problem in two separate steps where we first identify the blur parameters at each pixel $\boldsymbol{H_{d,m}}$ (see Sec. 7.3) and then estimate the sharp image $\boldsymbol{f}$ (see Sec. 7.5).

Notice that the above model works with any type of lens aperture. Therefore, we look for an aperture that allows a good reconstruction of both $\boldsymbol{f}$ and blur. We find that solving the above problem for conventional compact aperture yields poor results (see, for example, Table 7.1 in Sec. 7.6) due to a poor identifiability of the blur parameters and a stronger degradation of the image $\boldsymbol{f}$ (Sec.7.4). Our analysis shows that if the aperture is instead fragmented, both the parameter identification and the image degradation improve not only with still images, as already shown in [50, 42, 30], but also with motion-blur.

7.3 Motion and Depth Estimation

For now, assume that an aperture is given. As we consider a local patch of the input image and use constant velocity motion and constant defocus assumption, we can look for a blur identification method that does not require the simultaneous estimation of the sharp image $\boldsymbol{f}$. A successful method in blind deconvolution is the projection onto subspaces [27]. The key idea is that instead of solving problem (7.3) one minimizes

$$\tilde{\boldsymbol{d}}, \tilde{\boldsymbol{m}} = \underset{\boldsymbol{d},\boldsymbol{m}}{\operatorname{argmin}} \left[||\boldsymbol{H}^{\perp}_{\boldsymbol{d},\boldsymbol{m}}\, \boldsymbol{g}||^2 + \beta||\nabla \boldsymbol{d}|| + \gamma||\nabla \boldsymbol{m}|| \right] \tag{7.4}$$

By solving this problem via subspace projections one can show that Gaussian priors on the unknown image $\boldsymbol{f}$ are implicitly used. This however, is not a severe limitation, as also noticed by [52]. For a given defocus scale d_i and motion m_i, the local kernel $\boldsymbol{H}^{\perp}_{d_i,m_i}$ is a collection of orthonormal vectors. The energy term corresponds to the projection of a patch of $\boldsymbol{g}$ to a subspace. The local kernel can be computed directly from the analytic forms of the blur kernels or learned from synthetic and real data as was shown in [27]. In our implementation we learn the local kernels by using real sharp images of size $\delta \times \delta$ synthetically motion-blurred and defocused. The

procedure is rather straightforward, as one simply needs to: (1) generate a training set $\boldsymbol{G_{d,m}} = [\boldsymbol{g}_1\ \boldsymbol{g}_2\ \ldots\ \boldsymbol{g}_M]^T \in \mathbb{R}^{\delta^2 \times M}$ of images blurred with a specific parameter choice, (2) perform its singular value decomposition $\boldsymbol{G_{d,m}} = USV^T$ where $U = [U_1\ U_2\ \ldots\ U_{\delta^2}] \in \mathbb{R}^{\delta^2 \times \delta^2}$, V are orthonormal matrices, and S is diagonal with the singular values of $\boldsymbol{G_{d,m}}$, (3) define $\boldsymbol{H}^{\perp}_{\boldsymbol{d,m}} = U_t$, with $t = 1, \ldots, T < \delta^2$.

These local kernels can then be used to perform the discrete minimization in equation (7.4) for all possible parameters via graph cuts [47]. Notice that the second and third terms are standard total variation penalization terms involving pairwise interactions between neighboring pixels.

7.4 Analysis of aperture fragmentation

In this section we devise analysis and a procedure to determine what aperture is most suitable for the purpose of motion and defocus deblurring. Similarly to [50], we perform a frequency analyses of the aperture, in order to find a fragmentation patter that allows to preserve more frequencies than the compact aperture for different motions and defocus scales.

7.4.1 Combinatorics of Aperture Fragmentation

Consider partitioning a conventional aperture in a regular grid and moving the partitions within the chosen grid. Fragmentation can be seen as an "n choose m" allocation where m holes are assigned among n possible locations. The number of possible combinations can be readily obtained as $\frac{n!}{m!(n-m)!}$, which grows rather quickly as we increase the number of partitions. Hence, exhaustive search for the optimal fragmentation becomes rapidly impractical. Fortunately, as seen in Section 3.3.1, diffraction poses a limit to the number of possible partitions by introducing a lower bound on the smallest diameter that we can consider before blur starts increasing rather than decreasing. By using equation (3.21) we can obtain the smallest size of each opening r_{min} such that a point will be reproduced clearly on the image sensor. We have that

$r_{min} \approx 2.79\ mm$.

By imposing that compact aperture and fragmented aperture must allow the same incoming light, one obtains that the maximum number m of possible square apertures is

$$m = \left\lfloor \pi \left(\frac{F}{2F_\#} \right)^2 \frac{1}{r_{min}^2} \right\rfloor \tag{7.5}$$

where $\lfloor a \rfloor$ denotes rounding to the largest integer not exceeding a, and $F_\#$ is the F-number which indicates the size of the lens aperture. Conversely, given the number m of possible square apertures, one obtains that the side of each square must be

$$r_a = \sqrt{\frac{\pi}{m}} \frac{F}{2F_\#}. \tag{7.6}$$

Let us illustrate these formulas with two examples. If we fix the aperture of the conventional camera to, for instance, $F_\# = F/9$ (i.e., an aperture with diameter $5.6mm$ for a $50mm$ focal length lens), then the area of the aperture is $24.2mm^2$. In the fragmented aperture we aim at covering the same area with openings that have a minimum area of

$$area_{min} = r_{min} \times r_{min} = 7.8\ mm^2 \tag{7.7}$$

each; this yields that no more than 3 square holes are possible, and therefore a modest 84 combinations in a 3×3 grid. Vice versa, suppose that we use $F_\# = F/7.1$ and we are interested in allocating 3 square holes, then each square must have sides of $3.6\ mm$ (which is the dimension that we use in our experiments). Clearly, by using larger lens apertures, grids with more combinations are possible.

7.4.2 Frequency Analysis

Now that we have reduced the search space, we need to define a metric to compare different apertures and establish how much degradation they introduce. The analysis is carried out in the frequency domain of each fragmented aperture. A small patch of $\boldsymbol{f}$ (e.g., 64×64 pixels) is represented via the complex Fourier series:

$f(i_1, i_2) = \sum_{n_1,n_2} \hat{f}_{n_1,n_2} e^{j(n_1 i_1 + n_2 i_2)}$, where $\hat{f}_{n_1,n_2}$ are the Fourier coefficients, and i_1 and i_2 are pixels of the image. If we assume that the signal $\boldsymbol{f}$ is corrupted by additional noise bounded in absolute value by ω, we will not be able to recover frequencies corresponding to Fourier coefficients below the noise level. Hence, we can define the number of Fourier coefficients above a given noise level as a metric for the degradation introduced by a certain aperture across several motion blur and defocus scale parameters. If $\hat{k}^{d,m}_{n_1,n_2}$ denotes the Fourier coefficients of the blurring kernel $\boldsymbol{k}_{d,m}$, we can define the degradation metric M_ω as

$$M_\omega = \sum_{d,m} \sum_{n_1,n_2} (|\hat{k}^{d,m}_{n_1,n_2} \hat{f}_{n_1,n_2}| > w). \tag{7.8}$$

In comparing different apertures we fix $\hat{f} = 1$ at all frequencies and look for the highest M_ω. This analysis results in three optimal apertures shown in Fig. 7.2, where we have examined 10 noise levels for ω between 10^{-2} and 10^{-1}. In Fig. 7.2 we show 1D slices corresponding to noise levels $\omega = 0.04, 0.05, 0.1$ of the normalized 2D frequency domain, to illustrate that fragmentation better distributes degradation across the frequency domain. The frequency response of a conventional aperture (a disk with diameter $6.8mm$) is shown with a dashed red plot. We now consider fragmenting the conventional disk aperture in a collection of smaller apertures, thus retaining the same overall incoming light. All apertures have the same noise levels. The corresponding response of the three best fragmented apertures for each noise level are shown in solid blue and the noise level (constant across all frequencies) is shown in solid green. We find that fragmentation results in more frequencies above the given noise level.

7.5 Space-Varying Deblurring

Given the blur parameters provided by the procedure in Sec. 7.3, the space-varying deblurring task is a simpler problem. Indeed, the image formation model is linear in the unknown sharp image (although not a convolution) and efficient and stable

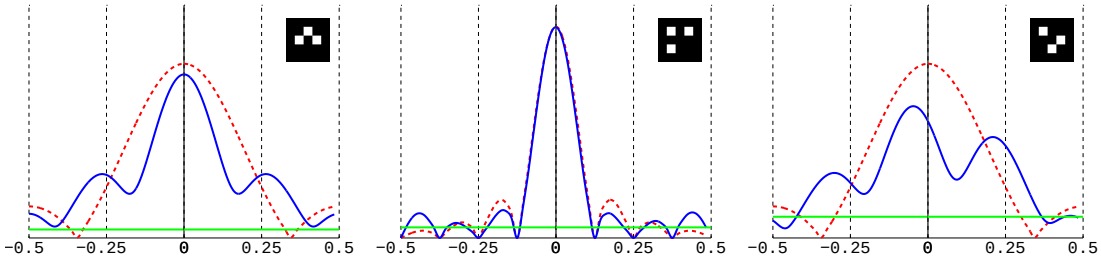


Figure 7.2: Frequency analysis of aperture patterns. The dashed red graph corresponds to different 1D slices of the frequency response of a compact aperture, while the solid blue corresponds to a fragmented aperture; the green threshold indicates the noise level. Left: Best fragmentation (evaluated over the entire 2D spectrum, not just a 1D slice) for noise levels $\omega = 0.01 - 0.04$. Middle: Best one for noise levels $\omega = 0.05 - 0.08$. Right: Best one for noise levels $\omega = 0.09 - 0.10$.

schemes for piecewise constant regularization exist.

As a first step we compute the first-order variation of the cost functional in equation (7.3) and obtain a discrete linearized version of the Euler-Lagrange equations

$$\boldsymbol{H}_{\tilde{\boldsymbol{d}},\tilde{\boldsymbol{m}}}^{T}\left(\boldsymbol{g}-\hat{\boldsymbol{g}}\right)+\alpha\,\boldsymbol{C}\cdot\boldsymbol{f}=0, \tag{7.9}$$

where $\boldsymbol{C}$ is a matrix operator which performs a discretization of $\boldsymbol{f}$ based on the previous estimate, as described in [12]. As we reduced the cost functional minimization in equation (7.3) to solving a linear system, standard numerical solvers can be used. Unfortunately, because the linear system involves blur, it is not diagonally dominant and fast solvers such as Gauss-Seidel or successive overrelaxation cannot be employed. We resort to conjugate gradient descent which does not have such limitations: It converges in a finite number of steps and it is fairly efficient ($\sim$ 1 minute for a 640×480 image with a Matlab implementation under a MacPro 2.6GHz quad-core CPU).

Mask	Mean error / Accuracy		
	defocus scale	motion direction	motion size
Disk	2.53 / 13.5%	0.22 / 88.8%	1.48 / 27.2%
Pattern A	0.41 / 80.8%	0.15 / 94.0%	1.48 / 31.0%
Pattern B	0.63 / 77.3%	0.24 / 89.5%	1.53 / 30.8%
Pattern C	**0.35 / 85.0%**	**0.11 / 95.0%**	**1.39 / 33.2%**

Table 7.1: Aperture performance.

7.6 Experiments

7.6.1 Performance

Before testing our algorithm with real images, we run a simulation to compare the performance of both defocus and motion estimation with the conventional aperture and the optimal patterns we have found in the frequency analysis (Sec. 7.4.2). The performance is evaluated under the same overall aperture incoming light over a set of 10 possible depth (defocus) and 64 different motions (8 directions by 8 sizes). For each level we take the same image (70x70 pixels) of random texture and we simulate both the defocus process and the motion blur using equation (7.1). Then we apply the local kernel, learnt as described in Sec. 7.3, and obtain a blur estimation (defocus scale, motion direction, and motion size). The output of the algorithm is then compared with the groundtruth in order to compute the error at each pixels. Table 7.1 reports the mean error and the accuracy (percentage of correct estimated pixels) for each type of aperture: the first one (disk) is the aperture of a conventional camera, while masks A, B, and C are the patterns shown in Fig. 7.2 from top to bottom respectively. Notice that the fragmented apertures can reach an higher performance than a compact aperture.

7.6.2 Real Data

We capture real images with the aperture pattern *C* (the rightmost pattern in Fig. 7.2) as it gives the best performance in the synthetic analysis. The size of each of the 3 square apertures is $3.6mm$, which corresponds to a compact (circular) aperture of about $7mm$ diameter (F/7.1 with a Canon $50mm$ lens). In Fig. 7.3 we captured a typical

(a) Coded image (b) Sharp image (from compact aperture)

(c) Defocus deblurring only (from coded aperture) (d) Sharp image (from coded aperture)

Figure 7.3: Results on real data for motion and defocus deblurring. (a) Input image when using the fragmented aperture (pattern C); (b) sharp image obtained by applying the method presented here to Figure 7.1; (c) estimated image when *only* defocus blur is removed; (d) sharp image when *both* defocus and motion blurs are removed.

scenario (the same picture with the relative compact aperture is shown in Fig. 7.1), where the camera brings into focus an area close to the foreground (the red bus in this scene), leaving the shops in the background out-of-focus. At the same time, the bus is moving from right to left, while the rest of the scene is still. The maximum motion blur magnitude in this dataset is ~ 12 pixels. We show the reconstructed sharp texture when only defocus blur is removed and when both defocus and motion-blur are corrected.

7.7 Summary

The task of deblurring a single image degraded by space-varying motion blur and defocus is extremely ill-posed: a small variation in the data (for instance, due to noise) results in large variations of the blur parameters and the restored image. It shown that blur parameters and details of the original sharp image can be recovered more easily if

one considers a coded aperture instead of a conventional aperture lens. Based on this analysis an algorithm has been proposed, where blur parameters are first identified by using local projections onto subspaces and deblurring is then performed as a separate step given all the blur parameters. This procedure is then successfully tested on a real scenario. Although a parametric representation of motion is considered, it is believe that this is the first solution for a space-varying deblurring algorithm from a single image.

This page has been left intentionally blank.

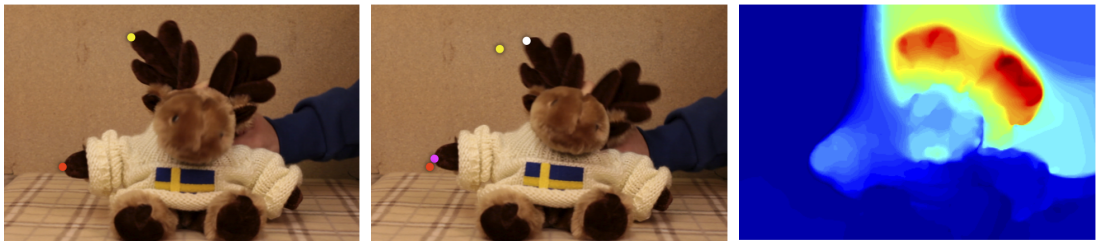
Chapter 8

Depth from a Video with Moving and Deformable Objects

It is not length of life, but depth of life.
Ralph Waldo Emerson [1803-1882]

Similarly to the previous chapter, a scene with moving and deformable objects is considered, but this time the depth is estimated from a monocular video sequence. While common techniques based on single camera view (*e.g.*, optical flow) successfully estimate depth in a rigid scene, they fail when there is a deformable scene (see Figure 8.1). Since no information about the motion and the deformation of the objects is provided, one cannot rely on matching multiple frames; instead one must rely on the information available in each single frame.

The depth estimation algorithm is based on the approach for general patterns (Chapter 6). However, an approximation of the method is implemented here in order to have a reasonably fast algorithm for processing several frames (Section 8.3). Section 8.2.3 introduces a novel spatial and temporal depth smoothness constraint, based on nonlocal-means (NLM) filtering, i.e., pixels whose intensities match within windows and within neighbouring frames are likely to share similar depths. Finally, in Section 8.4 the algorithm is successfully tested in real and challenging scenarios .



(a) Frame t-1 (b) Frame t (c) Optical flow

Figure 8.1: Optical flow in a non-rigid scenario. (a)-(b) Neighborhing frames from a video sequence, used as input for the optical flow estimated in (c). Notice that the resulting optical flow does not contain information about depth.

8.1 Related Work

Optical Flow and Structure from Motion. Depth estimation from a single video can be carried out in several ways, when the scene is rigid. The two most common techniques are optical flow and structure from motion. The former technique consists on finding correspondences between neighbouring frames and measuring the difference of their position: The shift is related to the depth of the scene only when the camera is moving and the scene is rigid [57, 90]. Models for non-rigid structures have been proposed in structure from motion [95, 96, 108], but they assume that feature correspondences are known [108] or occluders are treated as outliers [95, 96, 105] and then not reconstructed. Instead, the approach presented in this chapter estimates the depth of the whole scenario, including possible occluders. High-quality depth maps have been obtained in [104] from a video sequence captured with a freely moving camera. However, the method fails when moving or deformable objects are present in most of the area of the scene.

Nonlocal-Means Filters. To regularize the estimation, the concept of non-local mean filters is applied to depth reconstruction: The main idea is to link the depth values of pixels sharing the same colour (or texture). The concept of correlating pixels with similar colour or texture has been shown to be particularly effective in preserving edges in stereopsis [70, 87, 92] and thin structure in depth estimation [25, 90], as well

as image denoising [13, 82, 94].

8.2 Depth Estimation from Monocular Video

When one brings a part of the scene into focus, objects placed at different location appear out-of-focus; the amount of defocus depends on their location in the scene: More precisely, it depends on the distance between the objects and the focal plane. Because of this relationship, if we can identify the blur kernel for each object point in the scene, we can reconstruct the relative depth of the items in the scene. The exact distance from the camera can also be recovered from the blur size with a calibration procedure, once the camera setting is known.

In this section, the depth estimation algorithm is presented. The input is a video sequence captured by a single coded aperture camera. The scene is composed by object that moves independently: Therefore, one cannot rely on matching multiple frames, but has to extract as much depth information as possible at each single frame.

8.2.1 Imaging Formation Model

When we capture a video with a coded aperture camera, we have a set of T coded frames $\boldsymbol{g}_1, \boldsymbol{g}_2, \ldots, \boldsymbol{g}_T$. For each of these frames, the 3D scene, captured at a particular time t, can be decomposed in two entities: a 2D *sharp* frame $\boldsymbol{f}_t$, whose texture is all-in-focus, and a depthmap $\boldsymbol{d}_t$, which assigns a depth value (distance from the camera) to each pixel in $\boldsymbol{f}_t$. Our aim is to recover the geometry $\boldsymbol{d}_t$ of the scene at each time instant t. As described previously, different depths corresponds to different blur size in the coded image $\boldsymbol{g}_t$. Hence, the blur kernel $\boldsymbol{h}_p$, also called Point Spread Function (PSF), must be allowed to vary at each pixel p. If we consider all the elements ordered

as column vectors, we can write $\boldsymbol{g}_t$ as a product of matrices

$$\boldsymbol{g}_t = \underbrace{\begin{bmatrix} \boldsymbol{h}_1 & \boldsymbol{h}_2 & \dots & \boldsymbol{h}_N \end{bmatrix}}_{\boldsymbol{H}_{d_t}} \cdot \left.\begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_N \end{bmatrix}\right\} \boldsymbol{f}_t, \tag{8.1}$$

where N is the number of pixels of each frame and $\boldsymbol{H}_{\boldsymbol{d}_t}$ is a symmetric and sparse matrix that contains the information about the depth of the scene.

Since the scene is non-rigid (hence cannot rely on matching multiple frames), and since the sharp frames $\boldsymbol{f}_t$ are also unknown, in principle we should estimate both depth and all-in-focus image simultaneously from $\boldsymbol{g}_t$. However, it has been proved in [27] that this problem can be divided and solved in two separate steps: 1) depth estimation only and 2) image deblurring by using the estimated depth. In this paper, we focus our work on the former step.

We formulate the problem of depth estimation as a minimization of the cost functional

$$\hat{\boldsymbol{d}} = \underset{\boldsymbol{d}}{\operatorname{argmin}} \; E_{data}[d] + \alpha_1 E_{tv}[d] + \alpha_2 E_{nlm}[d] \; , \tag{8.2}$$

where α_1 and α_2 are two positive constants. In our approach, the data fidelity term $E_{data}[d]$ is taken from depth from single image (see Section 8.2.2) and we concentrate more on designing the regularization terms (Section 8.2.3).

8.2.2 Data Fidelity Term: Depth from Single Frame

The first term is based on the approach described in Chapter 6. The method identifies the blur size (and therefore the depth) at each pixel of a coded image by using projection onto subspaces. In our case, the depth $\boldsymbol{d}_t$ can be extracted from the single frame $\boldsymbol{g}_t$ without deblurring the image $\boldsymbol{f}_t$, by minimizing

$$E_{data}[d] = \sum_{\mathbf{p}} ||\boldsymbol{H}^{\perp}_{d_t(\mathbf{p})} \tilde{\boldsymbol{g}}^{\mathbf{p}}_t||^2_2 \tag{8.3}$$

where $\tilde{\boldsymbol{g}}_t^{\mathbf{p}}$ indicates the patch of size $\delta \times \delta$ centred at the pixel $\mathbf{p}$ at time t, that has been rearrange as a column vector. The symbol δ denotes the size of the maximum level of defocus considered. The matrix $\boldsymbol{H}_{\boldsymbol{d}_t}^{\perp}$ is built via a learning procedure, described in detail in Section 6.2.1, for each depth level d such that

$$\boldsymbol{H}_{d_i}^{\perp}\, \boldsymbol{H}_{d_j} \begin{cases} \approx 0, & \text{if } d_i = d_j \\ \gg 0, & \text{if } d_i \neq d_j \end{cases} \tag{8.4}$$

for any possible sharp texture $\boldsymbol{f}_t$.

Remarkable is the fact that, for depth estimation purpose only, there is no need to know the shape of the mask: In fact, the learning is performed on real coded images of a planar plane (with texture), placed at different distances from the camera.

Since we are processing videos, in Section 8.3.1 we work out possible solutions to approximate equation (8.3) in order to increase the efficiency of this algorithm and make it suitable for parallel computation.

8.2.3 Total Variation and Non-Local Means Filtering

The first regularization term $E_{tv}[d]$ in equation (8.2) represents the total variation

$$E_{tv}[d] = \int \|\nabla d(\mathbf{p})\|\, d\mathbf{p}\ , \tag{8.5}$$

which constrains the solutions to be piecewise constant [15]. However, this term alone tends to misplace the edge location and to remove thin surfaces, since it can combine together pixels that do not belong to the same surface.

To contrast this behaviour, we design a term that links depth values of pixels sharing the same colour (or texture) $E_{nlm}[d]$. Corresponding pixels can belong either to the same frame (Section 8.2.3.1) or to different frames (Section 8.2.3.2).

8.2.3.1 Spatial Smoothness

In this section we briefly analyzed how neighbourhood filtering methods establish correspondences between pixels and then extend the concept to our video sequence.

Many depth estimation methods assume that pixels with the same color or texture are likely to share also the same depth value. This can be obtained with a non-local *sigma*-filter [48], based on intensity differences

$$W_1(\mathbf{p},\mathbf{q}) = e^{-\frac{|g(\mathbf{p})-g(\mathbf{q})|^2}{\tau_1}}, \tag{8.6}$$

where the weight assigned to $W_1(\mathbf{p},\mathbf{q})$ represents how strong is the link between $\mathbf{p}$ and $\mathbf{q}$; or in other words, how likely they are to be located at the same depth. The symbol τ_1 indicates the bandwidth parameter determining the size of the filter. Loosely speaking, pixels with values much closer to each other than τ_1 are linked together, while the ones with values much more distant than τ_1 are not.

This type of filter has been largely used for image denoising, although they create some irregularities at the edges and in uniform regions [14], probably due to the pixel-based matching being sensitive to noise: To reduce this effect, one could use region-based matching as in the *non-local means* filter [25]:

$$W_1(\mathbf{p},\mathbf{q}) = e^{-\frac{G_\sigma * |g(\mathbf{p})-g(\mathbf{q})|^2(0)}{\tau_1}} \tag{8.7}$$

where G is an isotropic Gaussian kernel with variance σ such that

$$G_\sigma * |g(\mathbf{p}) - g(\mathbf{q})|^2(0) = \int_{\mathbb{R}^2} G_\sigma(\mathbf{x}) |g(\mathbf{p}+\mathbf{x}) - g(\mathbf{q}+\mathbf{x})|^2 |d\mathbf{x}. \tag{8.8}$$

Now we have obtained a neighbourhood filter for combining pixel of the same frame. However, since we have multiple frames, we can extend the correspondences to multiple frames.

8.2.3.2 Temporal Smoothness

Objects do not move much between neighbouring frames and we can still find some correspondences (although the region may be deformed).

Consider a pixel $\mathbf{p}$ from a frame $\boldsymbol{g}_{t_0}$ (captured at time t_0). We can rewrite the filter in equation (8.7) in a more general form, where the pixel $\mathbf{q}$ is now free to belong to any frame $\boldsymbol{g}_t$ of the video sequence

$$W_1(\mathbf{p}, t_0, \mathbf{q}, t) = e^{-\frac{G_{\sigma *}|g_{t_0}(\mathbf{p}) - g_t(\mathbf{q})|^2(0)}{\tau_1}}, \tag{8.9}$$

which included the case when $t = t_0$. Indeed, when considering the frame $\boldsymbol{g}_{t_0}$, the probability to find the same objects (or part of them) in another frame $\boldsymbol{g}_t$ decays moving away from the time t_0. Hence, we can add a filter that implements this likelihood:

$$W_2(t_0, t) = e^{-\frac{|t - t_0|}{\tau_2}} \tag{8.10}$$

where τ_2 is the bandwidth parameter in the temporal domain. This parameter is very important in deciding the amount of frames to consider in the regularization.

We can now combine the spatial (equation (8.7)) and the temporal (equation (8.10)) filters together to obtain the final filtering weights

$$W(\mathbf{p}, t_0, \mathbf{q}, t) = e^{-\frac{|t - t_0|}{\tau_2}} \; e^{-\frac{G_{\sigma *}|g_{t_0}(\mathbf{p}) - g_t(\mathbf{q})|^2(0)}{\tau_1}}. \tag{8.11}$$

Notice that, when the temporal term considers only 2 frames, t_0 and t_1, the corresponding pixels given by $W(\mathbf{p}, \mathbf{q}, t_0, t_1)$ include the matchings obtained from optical flow.

Finally, we use the sparse matrix $W(\mathbf{p}, t_0, \mathbf{q}, t)$ to define our neighbourhood regularization term, so that pixels with similar colors are encouraged to have similar depths value, *i.e.*

$$E_{nlm}[\boldsymbol{d}] = \int \int W(\mathbf{p}, t_0, \mathbf{q}, t) \; (d_t(\mathbf{q}) - d_{t_0}(\mathbf{p}))^2 \; d\mathbf{q} \; dt. \tag{8.12}$$

where $\mathbf{p}$ and $\mathbf{q}$ represent any pixel in the video sequence.

The term E_{nlm} is quadratic in the unknown depth map $\boldsymbol{d}$ and therefore it can be easily minimized.

8.3 Implementation Details

In this section we first study the data fidelity term equation (8.2) and find a sound approximation to improve the efficiency of the proposed method (Section 8.3.1). Secondly, we describe the iterative approach we adopt to minimize the cost functional in equation (8.2) (Section 8.3.2).

8.3.1 Filters Decomposition for Parallel Computation

We focus now on the computation of the data term $E_{data}[d]$. This term can quickly generate a non-regularized depthmapth (also called *raw* depthmap), when $\alpha_1 = \alpha_2 = 0$ in equation (8.2)). In this section, the subscripts $(_t)$ are assumed but omitted for clarity; the patches $\tilde{\boldsymbol{g}}_t^{\mathbf{p}}$ will then be denoted as $\tilde{\boldsymbol{g}}_{\mathbf{p}}$.

Since $\boldsymbol{H}_{\boldsymbol{d}}^{\perp}$ is a projection, we can rewrite equation (8.3) as

$$E_{data}[d] = \sum_{\mathbf{p}} \tilde{\boldsymbol{g}}_{\mathbf{p}}^T \, \boldsymbol{H}_{d(\mathbf{p})}^{\perp} \, \tilde{\boldsymbol{g}}_{\mathbf{p}} \; . \tag{8.13}$$

The computation of this term is suitable for parallel computation, since we can obtain a depth value at each pixel $\mathbf{p}$, independently on the other depth. Also, we have that $\boldsymbol{H}_{\boldsymbol{d}}^{\perp} = \boldsymbol{U}_{\boldsymbol{d}} \, \boldsymbol{U}_{\boldsymbol{d}}^T$ for construction, as defined in equation (6.24). With this observations, equation (8.13) becomes more memory efficient:

$$E_{data}[d(\mathbf{p})] = \|\tilde{\boldsymbol{g}}_{\mathbf{p}}^T \, \boldsymbol{U}_{d(\mathbf{p})}\|. \tag{8.14}$$

When using equation (8.14) as fidelity term, a raw depthmap of size 500×600 pixels can be obtained in about 200 s.

We look now into the bunch of filters $\boldsymbol{U_d}$ to check if there are possible approximations that can be adopted. The matrix $\boldsymbol{U_d} = [\boldsymbol{u}_{1,d}\ \boldsymbol{u}_{2,d}\ \dots\ \boldsymbol{u}_{M,d}]$ has size $\delta^2 \times M$, and its columns are *orthonormal* filters (Section 6.2.1). Therefore, equation (8.14) can be thought as a series of 2D convolutions between the whole image $\boldsymbol{g}$ and each column of $\boldsymbol{U_d}$ (both reshaped to 2D). This is done for each depth level d: we can then say that, to estimate the depth map for each frame of the video sequence, we have to compute $M \times N_d$ 2D-convolutions, where N_d is the number of depth levels considered. Just to have an idea of the dimensions we are dealing with, in our experiments we have $M \simeq 150$ and $N_d = 30$.

Since the total numbers of filters we use for each mask is much bigger than the size of each filter itself ($\delta \times \delta$, with $\delta = 33$), we can express each orthonormal filter $\boldsymbol{u}_{k,d}$ as a linear combination of a common base $\boldsymbol{B}$:

$$\boldsymbol{u}_{k,d} = \underbrace{\begin{bmatrix}\boldsymbol{b}_1 & \boldsymbol{b}_2 & \dots & \boldsymbol{b}_L\end{bmatrix}}_{\boldsymbol{B}} \cdot \boldsymbol{a}_{k,d}, \tag{8.15}$$

where $\boldsymbol{a}_{k,d}$ is a column vector containing the coefficients for the k-th filter at the depth d. By substituting equation (8.15) in equation (8.14), we can rewrite the fidelity term as

$$E_{data}[d(\mathbf{p})] = \left\| \tilde{\boldsymbol{g}}_{\mathbf{p}}^T\ \boldsymbol{B}\ \boldsymbol{A}_{d(\mathbf{p})} \right\|. \tag{8.16}$$

with $\boldsymbol{A}_{d(\mathbf{p})} = [\boldsymbol{a}_{1,d}\ \boldsymbol{a}_{2,d}\ \dots\ \boldsymbol{a}_{M,d}]$.

Notice that with this formulation we have reduced the number of 2D convolutions to the number of columns of $\boldsymbol{B}$; in other words, the complexity corresponds to the number of vectors that compose the common base (in our experiments, there are about 200 vectors). The depth map at each frame (500×600 pixels) can now be estimated in about 4 seconds.

In the following two section we illustrate how to estimate the common base $\boldsymbol{B}$ and the matrix of coefficients $\boldsymbol{A}$. These steps have to be run once, just after the learning of $\boldsymbol{H_d^{\perp}}$ for a given mask.

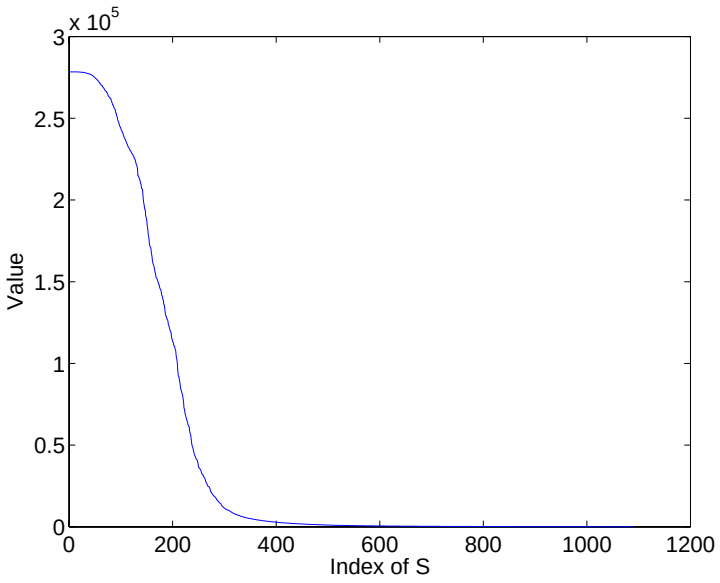


Figure 8.2: Eigenvalues of the matrix S in the SVD. The graph shows the values along the diagonal of $\boldsymbol{S}$; such values correspond to the eigenvalues of the matrix $\tilde{\boldsymbol{U}}$.

8.3.1.1 Estimating the common base B.

We build $\tilde{\boldsymbol{U}}$ (of size $\delta^2 \times M \times N_d$) by joining in the third dimensions the matrices $\boldsymbol{U}_d$ for all possible depth levels, $1 < d < N_d$. We then perform the singular value decomposition (SVD) of $\tilde{\boldsymbol{U}} = \boldsymbol{W}\boldsymbol{S}\boldsymbol{V}^T$: the most important *orthogonal* vectors that are in the left part of the matrix $\boldsymbol{W}$.

The diagonal of $\boldsymbol{S}$ contains the eigenvalues, i.e. the values that indicates the importance of each column of $\boldsymbol{W}$ to generate the space $\tilde{\boldsymbol{U}}$. The values along the diagonal are displayed with a graph in Figure 8.2.

The base $\boldsymbol{B}$ is then composed by the most important column of $\boldsymbol{W}$; experimentally, we have seen that the first 200 vectors are a good approximation for generating the space of $\tilde{\boldsymbol{U}}$.

8.3.1.2 Estimating the coefficients of the base.

Now that we have the common base $\boldsymbol{B}$, for each filter $\boldsymbol{u}_{k,d}$ we have to estimate the coefficients $\boldsymbol{a}_{k,d}$, such that equation (8.15) is satisfied. This yields to:

$$\boldsymbol{a}_{k,d}^T = \boldsymbol{u}_{k,d}^T \boldsymbol{B}^T (\boldsymbol{B}\boldsymbol{B}^T). \tag{8.17}$$

8.3.2 Iterative Linearization Approach

We solve the Euler-Lagrange equations of the cost functional in equation (8.2)

$$\nabla E[d] \doteq \nabla E_{data}[d] + \alpha_1 \nabla E_{tv}[d] + \alpha_2 \nabla E_{sm}[d] \tag{8.18}$$

via iterative linearization [12]. The second and third terms are can be computed easily as

$$\nabla E_{tv}[d] = -\nabla \cdot \left(\frac{\nabla d(\mathbf{p})}{|\nabla d(\mathbf{p})|} \right) \tag{8.19}$$

and

$$\nabla E_{nlm}[d] = \int \int W(\mathbf{p}, \mathbf{q}, t_0, t) \; (d(\mathbf{p}) - d(\mathbf{q})) \; d\mathbf{q} \; dt \tag{8.20}$$

while the data fidelity term requires a further analysis. In fact, the energy $E_{data}[d]$ has an irregular behaviour. Therefore, we expand our energy in Taylor series (stopping at the third term)

$$E_{data}[d] = E_{data}[\boldsymbol{d}_0] \; + \; \nabla E_{data}[\boldsymbol{d}_0](d - \boldsymbol{d}_0) \tag{8.21}$$

$$+ \frac{1}{2}(d - \boldsymbol{d}_0)^T \; \mathbf{H} E_{data}[\boldsymbol{d}_0](d - \boldsymbol{d}_0) \; , \tag{8.22}$$

where $\mathbf{H}$ indicates the *Hessian*. Now we can compute its derivative with respect to d

$$\nabla E_{data}[d] = \; \nabla E_{data}[\boldsymbol{d}_0] + \mathbf{H} E_{data}[\boldsymbol{d}_0](d - \boldsymbol{d}_0) \; , \tag{8.23}$$

where $\boldsymbol{d}_0$ represents the initial depth estimation obtained when setting $\alpha_1 = \alpha_2 = 0$.

Since the conditions for convergence require $\mathbf{H} E_{data}[\boldsymbol{d}_0]$ to be positive-definite, we

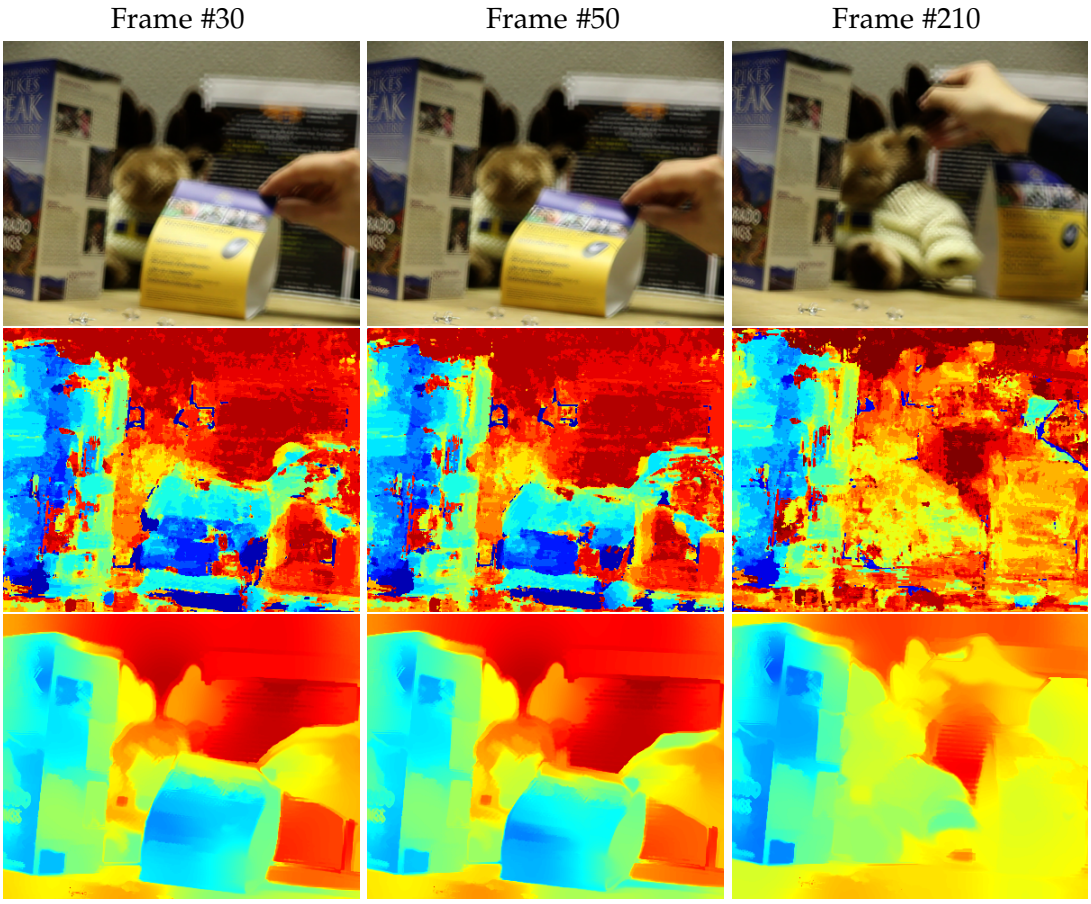


Figure 8.3: Depth estimation with objects deforming.. **Top row:** Some of the frames of the coded input video; **Central row:** Raw depth maps, estimated only with the data fidelity term, without any regularization ($\alpha_1 = \alpha_2 = 0$) ; **Bottom row:** Final depth maps obtained from our method.

consider instead $|\mathbf{H}E_{data}[\boldsymbol{d}_0]|$ and make it strictly diagonally dominant [103].

8.4 Experiments on Real Data

The videos have been captured by using a coded aperture camera, a Canon EOS-5D Mark-II with a mask inserted into a 50mm f/1.4 lens. The two datasets shown in this paper, Figure 8.3 and Figure 8.5, are very challenging scenario for depth estimation using a single camera. For both datasets, we show some coded frames from the video sequence and their relative depth maps that we have estimated. Below each input

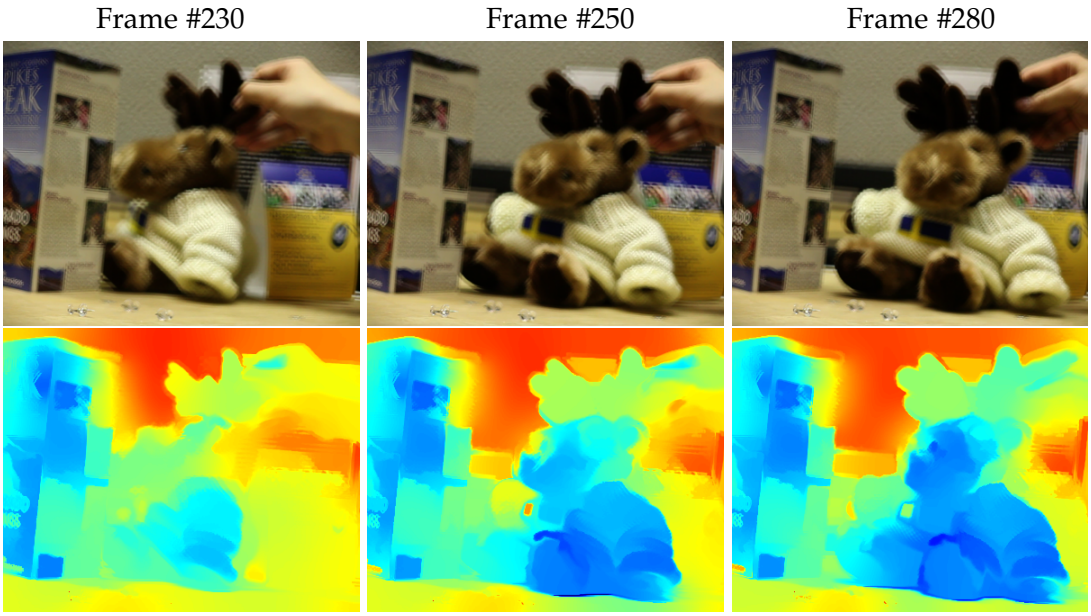


Figure 8.4: Depth estimation with objects moving. Top row: Some examples of frames from the coded input video; **Bottom row:** Depth maps reconstructed from our method.

frame there are two depth maps: 1) the raw depthmap (central row), obtained by minimizing only the term E_{data} and 2) the final depthmap (bottom row) resulting from minimizing the cost in equation (8.2).

Both videos have been taken with the camera "in hands", therefore the camera is also moving. The depth estimation, however, it is not affected by this shake. The video shown Figure 8.5 has been captured indoor in a very low light condition; therefore the input video is very noisy (ISO 2000). Nevertheless, the method still outputs impressive results, proving its robustness and consistency. Moreover, the quality of the results in this dataset may suggest that they can be used for tasks such as body pose estimation, or body part recognition.

8.5 Summary

A method to estimate depth from a single video with moving and deformable objects is presented for the first time. The approach is based on coded aperture technology, where a mask is placed on the lens of a conventional camera. Firstly, there is a deep

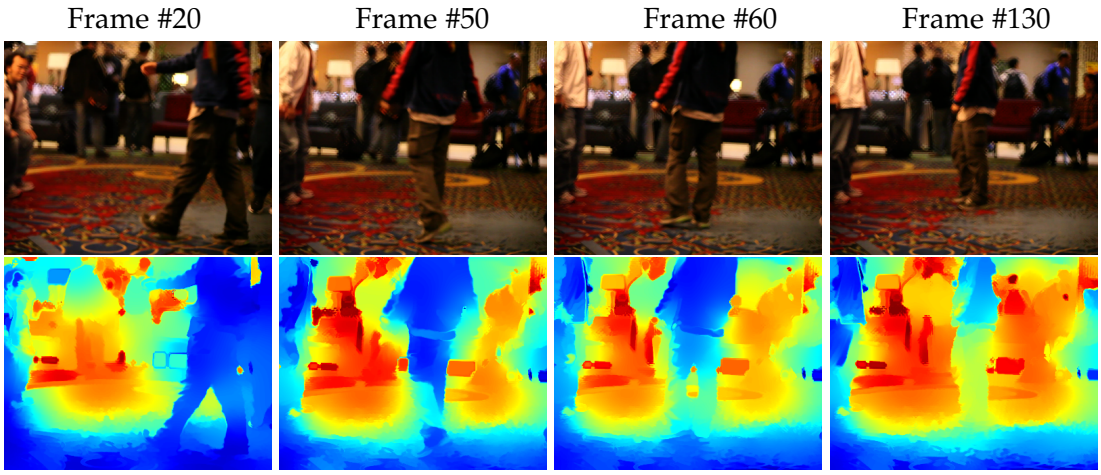


Figure 8.5: People Dataset. **Top row:** Some examples of frames from the coded input video; **Bottom row:** Depth maps reconstructed from our method.

analysis of the single image depth estimation method for general patterns in order to improve its efficiency; this is essential if dealing with video sequences. Secondly, a regularization term based non-local means filtering is introduced. This term creates at the same time a spatial and temporal neighbourhood of pixels that are likely to share the same depth value. The method is then tested on real data and high-quality depth maps are obtained from very challenging scenarios.

This page has been left intentionally blank.

Chapter 9

Coded Aperture Selection

One more thing.

Steve Jobs [1955-2011]

This chapter exploits a geometric interpretation of the blur, based on subspaces, which will be used to develop a mask selection criterion.

By using the formulation derived in Chapters 6, one can think blurred patches to be elements of different subspaces, where each subspace is characterised by a specific blur scale (Section 9.1). In this context, the procedure of blur estimation can be described as establishing the closest subspace to a given blurred patch. Ideally, one would like to have these subspaces as separate as possible from each other, in order to better distinguish the blur scale. However, the distances between the subspace changes depending on the pattern of the blur, which is in turn given by the aperture mask. Section 9.2 describes a possible metric to measure distances between subspaces and discusses how to obtain an optimal pattern for the purpose of depth and all-in-focus image reconstruction. All the aperture masks presented in literature, and used in this work, are tested under this criterion.

Moreover, in Section 9.3 the structures of asymmetric and symmetric aperture masks are investigated to comprehend wether one can distinguish the blur generated before and after the focal plane.

9.1 A Geometric Viewpoint on Blur Scale Identification

Chapter 6 has shown the blur scale at each pixel can be obtained by minimizing equation (6.25): One has to search among matrices $\boldsymbol{H}_{d_1}^{\perp}, \ldots, \boldsymbol{H}_{d_L}^{\perp}$ the one that yields the minimum ℓ_2 norm when applied to the vector $\boldsymbol{g}_{\mathbf{p}}$. This has a geometrical interpretation: Each matrix $\boldsymbol{H}_{d_i}^{\perp}$ defines a subspace and $\|\boldsymbol{H}_{d_i}^{\perp}\boldsymbol{g}_{\mathbf{p}}\|_2^2$ is the distance of each vector $\boldsymbol{g}_{\mathbf{p}}$ from that subspace.

Recall that $\boldsymbol{H}_{d_i}^{\perp} = \boldsymbol{U}_{\boldsymbol{d}_i}\boldsymbol{U}_{d_i}^T$ and that $U_i = [\boldsymbol{Q}_{d_i}\ \boldsymbol{U}_{d_i}]$ is an orthonormal matrix. Then the data term in equation (6.25) can be written as

$$\|\boldsymbol{H}_{d_i}^{\perp}\boldsymbol{g}_{\mathbf{p}}\|_2^2 = \|\boldsymbol{U}_{d_i}\boldsymbol{U}_{d_i}^T\boldsymbol{g}_{\mathbf{p}}\|_2^2 = \|\boldsymbol{U}_{d_i}^T\boldsymbol{g}_{\mathbf{p}}\|_2^2 = \|\boldsymbol{g}_{\mathbf{p}}\|_2^2 - \|\boldsymbol{Q}_{d_i}^T\boldsymbol{g}_{\mathbf{p}}\|_2^2. \tag{9.1}$$

Equation (9.1) can now be divided by the scalar number $\|\boldsymbol{g}_{\mathbf{p}}\|_2^2$: This yields exactly to the square of the subspace distance [91]

$$\mathcal{M}(\boldsymbol{g}, \boldsymbol{Q}_{d_i}) = \sqrt{1 - \sum_{j=1}^{K}\left(Q_{d_i,j}^T\frac{\boldsymbol{g}}{\|\boldsymbol{g}\|}\right)^2}, \tag{9.2}$$

where K is the rank of the subspace $\boldsymbol{Q}_{d_i}$, $\boldsymbol{Q}_{d_i} = [Q_{d_i,1}\ \ldots\ Q_{d_i,K}]$, and $Q_{d_i,j}$, $j = 1, \cdots, K$ are orthonormal vectors.

The geometrical interpretation brings a fresh look to image blurring and deblurring. Consider the image model (3.27), where a blurred image $\boldsymbol{g}$ is generated by multiplying a sharp image $\boldsymbol{f}$ on the right by a matrix $\boldsymbol{H}_{\boldsymbol{d}}$. The singular value decomposition of the blur matrix $\boldsymbol{H}_{\boldsymbol{d}}$ is given by

$$\boldsymbol{H}_{\boldsymbol{d}} = U_{\boldsymbol{d}}S_{\boldsymbol{d}}V_{\boldsymbol{d}}^T \tag{9.3}$$

where $S_{\boldsymbol{d}}$ is a diagonal matrix with positive entries, and both $U_{\boldsymbol{d}}$ and $V_{\boldsymbol{d}}$ are orthonormal matrices. Formally, the vector $\boldsymbol{f}$ undergoes a rotation ($V_{\boldsymbol{d}}^T$), then a scaling ($S_{\boldsymbol{d}}$), and then again another rotation ($U_{\boldsymbol{d}}$) (see Figure 9.1). This means that if $\boldsymbol{f}$ lives in a subspace, the initial subspace is mapped to another rotated subspace, possibly of

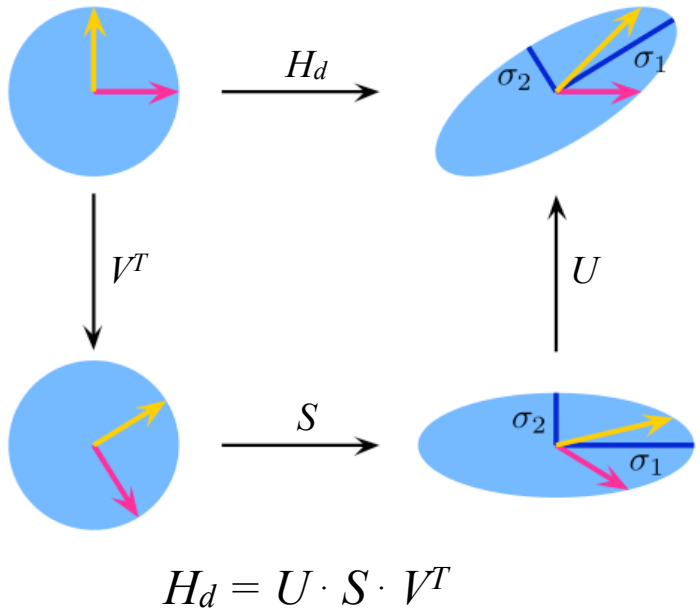
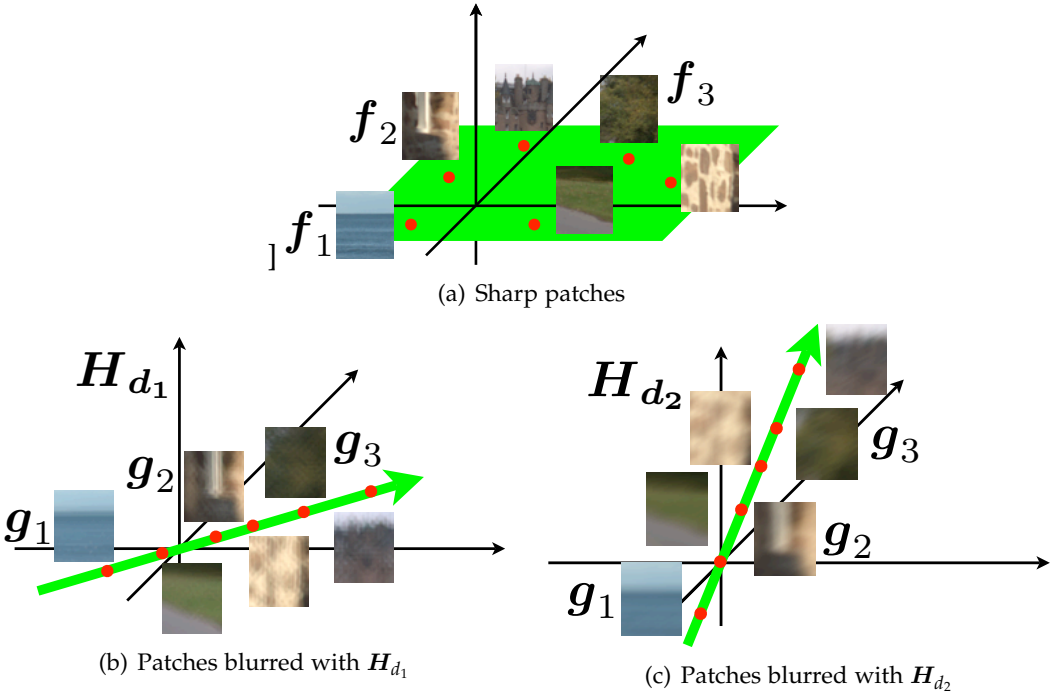


Figure 9.1: Geometric interpretation of SVD. Based on the singular value decomposition of $\boldsymbol{H}_d$, the effect of the blur on a sharp image is described as a rotation ($\boldsymbol{V}^T$), a scaling ($\boldsymbol{S}$), and a second rotation ($\boldsymbol{U}$).

smaller dimension (see Figure 9.2(b)). Notice that as the blur scale changes, the rotations and scaling are also changing and this may result in yet a different subspace (see Figure 9.2(c)).

It is important to understand that rotations of the vector $\boldsymbol{f}$ can result in blurring. To clarify this, consider blurred and sharp images with only 3 pixels (we cannot visualize the case of more than 3 pixels), *i.e.*, $\boldsymbol{g}_1 = [\boldsymbol{g}_{1,x}\ \boldsymbol{g}_{1,y}\ \boldsymbol{g}_{1,z}]^T$ and $\boldsymbol{f}_1 = [\boldsymbol{f}_{1,x}\ \boldsymbol{f}_{1,y}\ \boldsymbol{f}_{1,z}]^T$. Then, one can plot the vectors $\boldsymbol{g}_1$ and $\boldsymbol{f}_1$ as 3D points (see Figure 9.2). Let $\|\boldsymbol{g}_1\| = 1$ and $\|\boldsymbol{f}_1\| = 1$. Then, $\boldsymbol{f}_1$ can be rotated aroud the origin and overlap it exactly on $\boldsymbol{g}_1$. In this case rotation corresponded to blurring. The opposite is also true. The vector $\boldsymbol{g}_1$ can be rotated onto the vector $\boldsymbol{f}_1$ and thus perform deblurring. Furthermore, notice that in this simple example the most blurred images are vectors with identical entries. Such blurred images lie along the diagonal direction $[1\ 1\ 1]^T$. In general, blurry images tend to have entries with similar values and hence tend to cluster around the diagonal direction.



(a) Sharp patches

(b) Patches blurred with H_{d_1}

(c) Patches blurred with H_{d_2}

Figure 9.2: Coded images subspaces. (a) Patches of sharp images on a subspace. (b) Subspace containing images blurred with H_{d_1}; blurring has the effect of rotating and possibly reducing the dimensionality of the original subspace. (c) Subspace containing images blurred with H_{d_2}.

The ability to discriminate between different blur scales in a blurry image boils down to being able to determine the subspaces where the patches of such blurry image live. If sharp images do not live on a subspace, but uniformly in the entire space, the only way to distinguish the blur size is that the blurring $\boldsymbol{H_d}$ scales some dimensions of $\boldsymbol{f}$ to zero and that the scaling varies with blur size. This case has links to the zero-sheet approach in the Fourier domain [74]. However, if the sharp images live on a subspace, the blurring $\boldsymbol{H_d}$ may preserve all the directions and blur scale identification is still possible by determining the rotation of the sharp images subspace. This is the principle that is exploited here.

Notice that the evaluation of the subspace distance $\mathcal{M}$ involves the calculation of the inner product between a patch and a column of $\boldsymbol{U_{d_i}}$. Hence, this calculation can be done exactly as the convolution of a column of $\boldsymbol{U_{d_i}}$, rearranged as an image patch,

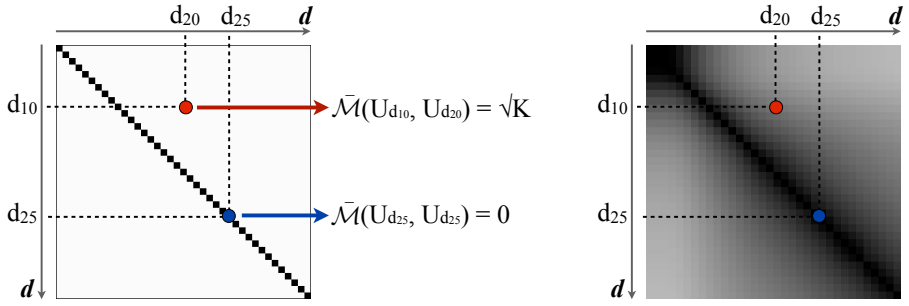
with the whole image $\boldsymbol{g}$. In conclusion, the algorithm requires computing a set of $L \times K$ convolutions with the coded image, which is a stable operation of polynomial computational complexity.

9.2 Coded Aperture Selection Criterion

This section discusses how to obtain an optimal pattern for the purpose of depth and all-in-focus image reconstruction. As pointed out in [21] there are two main challenges: The first one is that accurate estimation of depth and texture requires accurate identification of the blur scale; the second one is that accurate deblurring requires little texture loss due to blurring. A first step towards addressing these challenges is to define a metric for blur scale identification and a metric for texture loss. Our metric for blur scale identification can be defined directly from section 9.1. Indeed, the ability to determine which subspace a coded image patch belongs to can be measured via the distance between the subspaces associated to each blur scale

$$\bar{\mathcal{M}}(\boldsymbol{U}_{\boldsymbol{d}_1}, \boldsymbol{U}_{\boldsymbol{d}_2}) = \sqrt{K - \sum_{i,j} \left(U_{\boldsymbol{d}_1,i}^T U_{\boldsymbol{d}_2,j} \right)^2}. \tag{9.4}$$

Clearly, the wider apart all the subspaces are, and the less prone to noise the subspace association is. We find that a good visual summary of the "spacing" between all the subspaces is a (symmetric) matrix with distances between any two subspaces. We compute such matrix for a conventional camera and show the results in Figure 9.3, together with the ideal distance matrix. In each distance matrix, subspaces associated to blur scales ranging from the smallest to the largest ones are arranged along the rows from left to right and along the columns from top to bottom. Along the diagonal the distance is necessarily 0 as we compare identical subspaces. Also, by definition the metric cannot exceed $\sqrt{K}$, where K is the minimum rank among the subspaces. In Figure 9.5 we report the distance matrices computed for each of the apertures we consider in this work (see Figure 9.4).

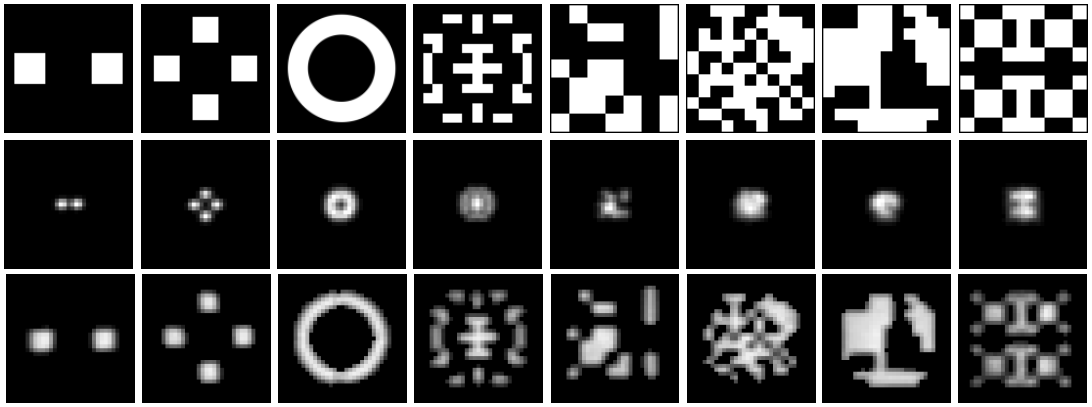


(a) Ideal distance matrix (b) Circular aperture

Figure 9.3: Distance matrix computation. The top-left corner of each matrix is the distance between subspaces corresponding to small blur scales, and, vice versa, the bottom-right corner is the distance between subspaces corresponding to large blur scales. Notice that large subspace distances are bright and small subspace distances are dark. The maximum distance ($\sqrt{K}$) is achievable when two subspaces are orthogonal to each other.

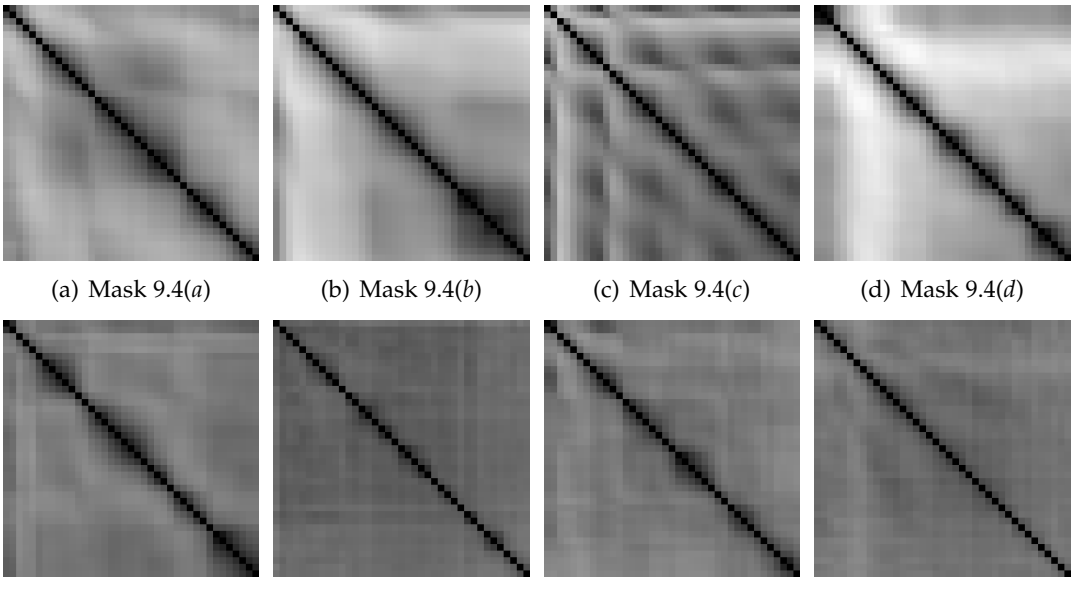
Notice that the subspace distance map for a conventional camera (Figure 9.3(b)) is overall darker than the matrices for coded aperture cameras (Figure 9.5). This shows the poor blur scale identifiability of the circular aperture and the improvement that can be achieved when using a more elaborate pattern.

The rank K can be used to address the second challenge, *i.e.*, the definition of a metric for texture loss. So far we have seen that blurring can be interpreted as a combination of rotations and scaling. Deblurring can then be interpreted as a combination of rotations and scaling in the opposite direction. However, when blurring scales some directions to 0, part of the texture content has been lost. This suggests that a simple measure for texture loss is the dimension of the coded subspace: The higher the dimension and the more texture content can be restored. As the (coded images) subspace dimension is K, one can immediately conclude that the subspace distance matrix that most closely resembles the ideal distance matrix (see Figure 9.3(a)) is the one that simultaneously achieves the best depth identification and the least texture loss. Finally, we propose to use the average L_1 fitting of any distance matrix to the ideal distance matrix scaled of $\sqrt{K}$, *i.e.*, $|\sqrt{K}(\mathbf{1}\mathbf{1}^T - \boldsymbol{I}) - \bar{\mathcal{M}}|$. The fitting yields the values in Table 9.1. We can also see visually in Figure 9.5 that mask 4.4(b) and mask 4.4(d) are the coded apertures that we can expect to achieve the best results in texture



(a) (b) (c) (d) (e) (f) (g) (h)

Figure 9.4: Coded aperture patterns and PSFs. All the aperture patterns we consider in this work (top row) and their calibrated PSFs for two different blur scales (second and bottom row). (a) and (b) aperture masks used in both [39] and [62]; (c) annular mask used in [64]; (d) pattern proposed by [50]; (e) pattern proposed by [98]; (f) and (g) aperture masks used in [106]; (h) MURA pattern used in [34].



(a) Mask 9.4(*a*) (b) Mask 9.4(*b*) (c) Mask 9.4(*c*) (d) Mask 9.4(*d*)

(e) Mask 9.4(*e*) (f) Mask 9.4(*f*) (g) Mask 9.4(*g*) (h) Mask 9.4(*h*)

Figure 9.5: Subspace distances for the eight masks in Figure 9.4. Notice that the subspace rank K determines the maximum distance achievable, and therefore, coded apertures with overall darker subspace distance maps have poor blur scale identifiability (*i.e.*, sensitive to noise).

	Masks							
	4.4(a)	4.4(b)	4.4(c)	4.4(d)	4.4(e)	4.4(f)	4.4(g)	4.4(h)
L_1 **fitting**	8.24	6.62	8.21	5.63	8.37	16.96	8.17	16.13

Table 9.1: L_1 **fitting** of any distance matrix to the ideal distance matrix scaled of $\sqrt{K}$.

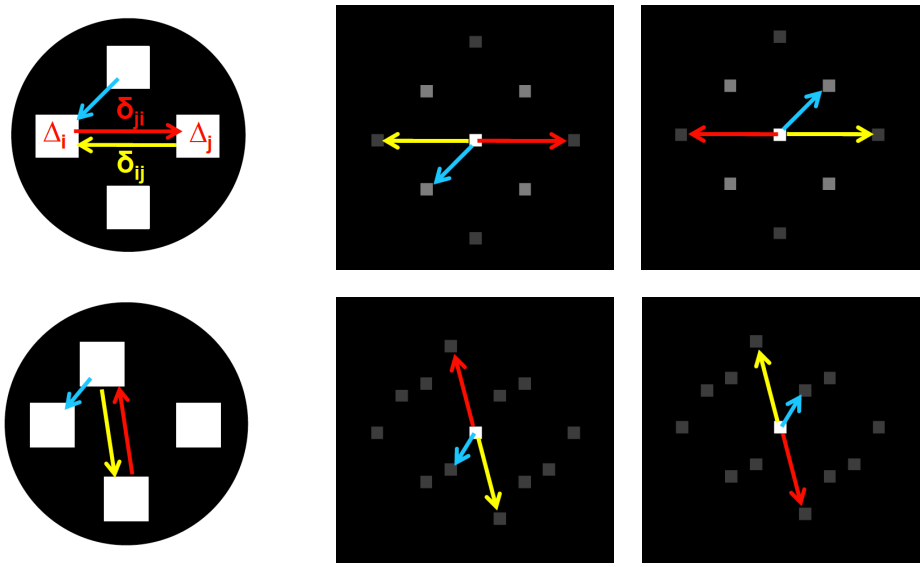
deblurring.

The quest for the optimal mask is, however, still an open problem. Even if we look for the optimal mask via brute-force search, a single aperture pattern requires the evaluation of equation (9.4) and the computation of all the subspaces associated to each blur scale. In particular, the latter process requires about 8 minutes on a QuadCore 2.8GHz with Matlab 7, which makes the evaluation of a large number of masks unfeasible.

9.3 Symmetric vs. Asymmetric Masks

This section presents an investigation on whether the choice of the aperture mask can be crucial to distinguish when an object is placed *before* from when it is placed *after* the focal plane. The image generated in the two cases can be easily simulated by using the image formation model described in Chapter 3. An object point placed between the camera and the focal plane generates a blur that has exactly the same shape than the aperture mask. Instead, an object point located after the focal plane generates a blur whose shape is a flipped version of the mask. Therefore, the first conclusion can be drawn: to distinguish the two sides of the scene, the aperture mask cannot be symmetric.

Nevertheless, there is a problem even with an asymmetric mask. To be able to distinguish the two blurs, one has to have access to the blur matrix $\boldsymbol{H}_d$, which contains the blur kernel d that generated the image $\boldsymbol{g}$. However, recalling the depth estimation methods previously presented in this work, one notice that $\boldsymbol{H}_d$ is always multiplied by its transpose, $\boldsymbol{H}_d^T$. The first approach (Chapter 5) is derived from the following



(a) mask (b) before focal plane (c) after focal plane

Figure 9.6: Before and after the focal plane. The structure of N_p described in Chapter 5 is base on the product $\boldsymbol{H}_d\boldsymbol{H}_d^T$. Such a structure is shown, for the same blur scale, before and after the focal plane for a symmetric (top) and an asymmetric mask (bottom).

definition of Σ_k (from equation (5.11))

$$\boldsymbol{\Sigma}_k(\boldsymbol{A}_k) = \boldsymbol{H}_d\boldsymbol{A}_k^{-1}\boldsymbol{H}_d^T + \boldsymbol{C}_k\sigma^2\boldsymbol{I}\ ; \tag{9.5}$$

The second approach (Chapter 6) is based on the bank of filters $\boldsymbol{H}_{\boldsymbol{d}}^{\perp}$, which is defined from equation (6.6) as

$$\boldsymbol{H}_d^{\perp} \doteq \boldsymbol{I} - \boldsymbol{H}_d\left(\alpha\boldsymbol{\Sigma}^T\boldsymbol{\Sigma} + \boldsymbol{H}_d^T\boldsymbol{H}_d\right)^{-1}\boldsymbol{H}_d^T\ . \tag{9.6}$$

To better illustrate the problem, we recall the structure of N_p, described in Chapter 5, which is base on the product $\boldsymbol{H}_d\boldsymbol{H}_d^T$. Figure 9.6 show such a structure, for the same blur scale, before and after the focal plane for a symmetric (top row) and an asymmetric (bottom row) mask. When the product is computed, there asymmetric property is lost and it is not possible to distinguish between a blur generated before and one generated after the focal plane.

9.4 Summary

This chapter presents the blur effect as a rotation and a scaling of subspaces. Based on this interpretation, blurred images lives in different subspaces, each of them characterised by a specific blur. For the benefit of blur identification, these subspaces should lie as far as possible from each other. Hence, a mask selection criterion has been developed to find the aperture pattern that maximises subspaces distances. However, given the current procedure of the mask selection criterion, the evaluation of a large number of masks is unfeasible. Devising a fast procedure to determine the optimal mask will be subject of future work.

The difference in using symmetric and asymmetric masks has also been investigated, in order to distinguish between the two sides of the scene divided by the focal plane. However this task requires further research, since the depth estimation methods presented here cannot solve this ambiguity.

This page has been left intentionally blank.

Chapter 10

Conclusions

The questions are always more
important than the answers.
Randy Pausch [1960-2008]

This thesis presented analysis and methods to address the problem of 3D scene reconstruction and all-in-focus image estimation from a single 2D image captured by a coded aperture camera. A coded aperture device consists of a conventional camera with a modified aperture mask placed on the main lens. With this device the depth information of the scene can be encoded in a single 2D image, to be later extracted and separated from the texture. The depth encoding is obtained by using the properties of defocus blur: Objects placed at different distances from the camera appear to have different blur scales in the captured image. The shape of the blur generated by out-of-focus objects takes the form of the lens aperture. The key advantage of using a coded aperture mask in the camera lens is that one can change the shape of the blur, such that it is more distinguishable from natural texture, and therefore easier to identify. When blur scale is identified, one can 1) extract the real depth value through a calibration procedure, 2) estimate the all-in-focus image, and also 3) combine the depth and the sharp image together to simulate stereoscopic vision (as illustrated with some anaglyphs in the thesis).

This work has analyzed the problem of depth and all-in-focus image reconstruction from a single 2D coded image, and has demonstrated that this problem can be split in two steps, without compromises or approximations: first depth estimation and then image deblurring. The latter step uses the estimate of the former one.

Regarding the first step, two approaches have been proposed, depending on the complexity of the pattern in the aperture mask. If the pattern can be decomposed as a small set of identical openings, one can estimate the depth by considering the contributions of just a few pixels instead of the whole patch, hence reducing drastically the computational complexity. If the pattern is more complex, the depth can be estimated with a bank of orthogonal filters previously learned. One of the main advantages of this method is that, for the purpose of depth estimation alone, it does not require to have any information about the shape of the aperture mask: The bank of filters can be learned directly from a set of blurred images. The second important advantage is that the learning procedure has to be carried out only once for a given mask. The range of distinguishable depth levels can then be easily re-mapped to any other scenario, using the image formation model described in the thesis.

The second part of the problem consists of recovering the all-in-focus image. This is achieved by using a deconvolution step with the estimated depth map. Its implementation is based on a gradient descent algorithm.

The proposed solution has also been extended to a non rigid scenario, where objects can deform and move independently. Under these circumstances, the algorithm can successfully remove the degradation created in an image by both defocus and motion blur. Although only a parametric representation of motion is considered, it is the first solution for a space-varying deblurring algorithm from a single image. Similarly, another problem has also been successfully addressed for the first time in the literature: depth reconstruction from a video sequence with moving and deformable objects in the scene. An import contribution of this approach is the introduction of a regularization term that creates, at the same time, a spatial and temporal neighbourhood of pixels that are likely to share the same depth value.

Results on both synthetic and real data are presented and compared to existing algorithms in the literature, showing that the proposed methods achieve state-of-the-art performance, without any user intervention. This big improvement is due to the fact that the proposed solutions bypass the reconstruction of the sharp image, allowing the method to deal with a larger amount of blur.

The quality of the results on real data and the robustness of the methods show the possibility for this technique to be used in applications that require depth estimation, such as obstacle avoidance. In addition, this technology can be adopted by applications that normally use only the texture information of an image, but that can gain accuracy from the additional 3D information; an example of such application is object detection. Furthermore, the quality of the results obtained from video sequences suggest that this technique may be very useful to tasks such as body pose estimation or body part recognition.

The next sections will discuss the drawbacks of this work, and how these could be addressed in future work.

10.1 Limitations of this Work

The first and most evident limitation of this approach is that the 3D information of the scene can be reconstructed only if the scene is *not* in-focus. Moreover, as already discussed in the Section 9.3, the proposed methods have an ambiguity: they cannot distinguish between *before* and *after* the focal plane. Therefore, the focal plane has to be set outside the scene of interest.

Another limitation in the depth estimation is the type of lens. Similarly to what happens in stereo, the depth resolution achievable with a coded aperture device is given by the size of the pattern, which is limited by the size of the lens aperture.

A minor limitation is that, when a binary mask is placed on a the lens of a conventional camera, it reduces the amount of light that goes through the lens, and therefore reduces the signal-to-noise ratio. However, this does not affect the proposed depth

estimation algorithms, which have been shown to be robust and achieve good results even in low-light conditions.

10.2 Future Work

The performance of the proposed approaches with different masks has shown that the design of an optimal mask is still an open problem. A mask selection criterion has been proposed in Chapter 9, but at the moment the evaluation of a large number of masks is unfeasible. Devising a fast procedure to determine the optimal mask will be subject of future work. Regarding the search for the optimal mask, further research is also required in order to distinguish between the two sides of the scene divided by the focal plane, since the depth estimation methods presented in this thesis cannot solve this ambiguity.

In the depth estimation method for general pattern, a bank of filters is learned from a set of blurred images. This is very useful for depth estimation, but for the all-in-focus image estimation one still needs to know the blur kernel. A very interesting work would be to identify the pattern of the aperture mask from the filters or directly from the blurred images.

The all-in-focus image estimation has been presented and implemented for a single image. However this can be extended to a video sequence. As for the depth estimation method, the quality of the deblurring results is expected to improve from the additional information regarding neighbouring frames.

Another approach to image reconstruction, which is not discussed in this work, is to estimate the sharp image that is "seen" at each opening of the mask. This would require 1) to include a model for occlusion in the image deblurring, 2) a very precise depth estimation at the edges.

References

[1] J. G. Ables. Fourier transform photography: a new method for X-ray astronomy. *Proceedings of the Astronomical Society of Australia*, 1:172–177, December 1968. 19

[2] Acidcow.com. Autostereogram (100 pics). Feb 2012. URL `http://acidcow.com/pics/1694-autostereogram-100-pics.html`. 6

[3] A. Agrawal and R. Raskar. Optimal single image capture for motion deblurring. *IEEE Conference on Computer Vision and Patter Recognition*, pages 2560–2567, Jun 2009. 98

[4] S. Bae and F. Durand. Defocus magnification. *Eurographics*, 26(3), 2007. 17

[5] M. Bertero and P. Boccacci. Introduction to inverse problems in imaging. *Institute of Physics Publishing, Bristol and Philadelphia*, 1998. 15, 80

[6] S. S. Bhasin and S. Chaudhuri. Depth from defocus in presence of partial self occlusion. *International Conference on Computer Vision*, 2:488–493, 2001. 31

[7] T. E. Bishop, R. Molina, and J. R. Hopgood. Blind restoration of blurred photographs via ar modelling and mcmc. *IEEE International Conference on Image Processing*, 1:669–672, Oct 2008. 15, 37

[8] T. E. Bishop, S. Zanetti, and P. Favaro. Light field superresolution. *IEEE International Conference on Computational Photography*, April 2009. 19

[9] M. Born and E. Wolf. *Principles of Optics*. Cambridge University Press, 1999. 25, 33

[10] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(9):1124–1137, September 2004. 15

[11] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(11): 1222–1239, November 2001. 50

[12] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert. High accuracy optical flow estimation based on a theory for warping. *European Conference on Computer Vision*, 4:25–36, May 2004. 104, 119

[13] A. Buades, B. Coll, and J.-M. Morel. A non-local algorithm for image denoising. *IEEE Conference on Computer Vision and Patter Recognition*, 2005. 111

[14] A. Buades, B. Coll, and J.-M. Morel. Nonlocal image and movie denoising. *International Journal of Computer Vision*, 76(2):123–139, 2007. 114

[15] T. F. Chan and J. Shen. *Image processing and analysis: variational, PDE, wavelet, and stochastic methods*. Society for Industrial and Applied Mathematics, 2005. 113

[16] G.K. Chantas, N.P. Galatsanos, A.C. Likas, and M. Saunders. Variational bayesian image restoration based on a product of t-distributions image prior. *IEEE Transactions on Image Processing*, 17(10):1795–1805, October 2008. 58, 59, 63

[17] S. Chaudhuri and A. Rajagopalan. *Depth from Defocus: a Real Aperture Imaging Approach*. Springer-Verlag, 1999. 29, 31, 33

[18] W. E. Crofts. *The Generation of Depth Maps via Depth-from-Defocus*. PhD thesis, University of Warwick, 2007. 18, 27, 28

[19] R.H. Dicke. Scatter-hole cameras for x-rays and gamma rays. *Astrophys Journal*, 153:101–112, August 1968. 19

[20] Q. Dou and P. Favaro. Off-axis aperture camera: 3d shape reconstruction and image restoration. *IEEE Conference on Computer Vision and Patter Recognition*, June 2008. 21, 32

[21] E. R. Dowski and T. W. Cathey. Single-lens single-image incoherent passive-ranging systems. *Applied Optics*, 33(29):6762–6773, 1994. 20, 128

[22] J. Ens and P. Lawrence. A matrix-based method for determining depth-from-defocus. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 600–606, 1991. 26

[23] J. Ens and P. Lawrence. An investigation of methods for determining depth from defocus. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 5(2): 97–108, 1993. 26

[24] H. Farid. *Range Estimation by Optical Differentiation*. PhD thesis, University of Pennsylvania, 1997. 16, 54

[25] P. Favaro. Recovering thin structures via nonlocal-means regularization with application to depth from defocus. *IEEE Conference on Computer Vision and Patter Recognition*, pages 1133 – 1140, Jun 2010. 110, 114

[26] P. Favaro and S. Soatto. Shape and radiance estimation from the information divergence of blurred images. *European Conference on Computer Vision*, 2000. 15

[27] P. Favaro and S. Soatto. A geometric approach to shape from defocus. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(3):406–417, Mar 2005. 16, 30, 76, 100, 112

[28] P. Favaro and S. Soatto. *3-D Shape Reconstruction and Image Restoration: Exploiting Defocus and Motion-Blur*. Springer-Verlag, 2006. 15, 28, 29, 31, 33

[29] E. E. Fenimore and T. Cannon. Coded aperture imaging with uniformly redundant rays. *Applied Optics*, 17(3):337– 347, 1978. 19

[30] R. Fergus, B. Singh, A. Hertzmann, S.T. Roweis, and W.T. Freeman. Removing camera shake from a single photograph. *ACM Trans. Graph.*, 25(3):787–794, Aug 2006. 98, 100

[31] T. Georgiev, K.C. Zheng, B. Curless, D. Salesin, S. Nayar, and C. Intawala. Spatio-angular resolution tradeoffs in integral photography. *Eurographics Workshop on Rendering*, pages 263–272, 2006. 19

[32] B. Girod and E. H. Adelson. System for ascertaining direction of blur in a range-from-defocus camera. *US Patent No. 4,939,515*, 1990. 18

[33] G. H. Golub and C. F. Van Loan. *Matrix computations*. The Johns Hopkins University Press, London, 3rd edition, 1996. 76

[34] S. R. Gottesman and E. E. Fenimore. New family of binary arrays for coded aperture imaging. *Applied Optics*, 28(20):4344–4352, October 1989. 53, 54, 80, 130

[35] P. Green, W. Sun, W. Matusik, and F. Durand. Multi-aperture photography. *ACM Trans. Graph.*, 26(3):68, 2007. 20

[36] J. Hadamard. Sur les problèmes aux dérivées partielles et leur signification physique. *Princeton University Bulletin*, pages 49–52, 1902. 48

[37] S. W. Hasinoff. *Variable-Aperture Photography*. PhD thesis, University of Toronto, 2008. 24, 30, 31

[38] S. W. Hasinoff and K. N. Kutulakos. A layer-based restoration framework for variable-aperture photography. *International Conference on Computer Vision*, pages 1–8, 2007. 15

[39] S. Hiura and T. Matsuyama. Depth measurement by the multi-focus camera. *IEEE Conference on Computer Vision and Patter Recognition*, 2:953–961, June 1998. 20, 53, 54, 67, 80, 130

[40] J. Huang and D. Mumford. Statistics of natural images and models. *IEEE Conference on Computer Vision and Patter Recognition*, 1:1541–1548, Jun 1999. 72

[41] J. Huang, A. Lee, and D. Mumford. Statistics of range images. *IEEE Conference on Computer Vision and Patter Recognition*, pages 324–331, Jun 2000. 73

[42] J. Jia. Single image motion deblurring using transparency. *IEEE Conference on Computer Vision and Patter Recognition*, pages 1–8, Jun 2007. 98, 100

[43] H. Jin and P. Favaro. A variational approach to shape from defocus. *European Conference on Computer Vision*, 2:18–30, 2002. 15

[44] G. E. Johnson, E. R. Dowski, and W. T. Cathey. Passive ranging through wavefront coding: Information and application. *Applied Optics*, 39(11):1700–1710, 2000. 20

[45] V. Kolmogorov and R. Zabih. Computing visual correspondence with occlusions via graph cuts. *International Conference on Computer Vision*, 2:508–515, 2001. 64

[46] V. Kolmogorov and R. Zabih. Multi-camera scene reconstruction via graph cuts. *European Conference on Computer Vision*, 3:82–96, 2002. 78

[47] V. Kolmogorov and R. Zabih. What energy functions can be minimized via graph cuts? *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(2): 147 – 159, Jan 2004. 101

[48] J. Lee. Digital image smoothing and the sigma filter. *Computer Vision, Graphics and Image Processing*, 24(2):255–269, Nov 1983. 114

[49] A. Levin. Blind motion deblurring using image statistics. *Advances in Neural Information Processing Systems*, pages 841–848, Dec 2006. 98

[50] A. Levin, R. Fergus, F. Durand, and W. T. Freeman. Image and depth from a conventional camera with a coded aperture. *ACM Trans. Graph.*, 26(3):70, Aug 2007. 3, 20, 49, 50, 51, 54, 55, 59, 74, 80, 87, 90, 91, 100, 101, 130

[51] A. Levin, P. Sand, T. S. Cho, F. Durand, and W. T. Freeman. Motion-invariant photography. *ACM Trans. Graph.*, 27(3), Aug 2008. 98

[52] A. Levin, Y. Weiss, F. Durand, and W. T. Freeman. Understanding and evaluating blind deconvolution algorithms. *IEEE Conference on Computer Vision and Patter Recognition*, pages 1964–1971, Jun 2009. 98, 100

[53] M. Levoy, R. Ng, A. Adams, M. Footer, and M. Horowitz. Light field microscopy. *ACM Trans. Graph.*, 25(3):924–934, Jul 2006. 19

[54] C.-K. Liang, T.-H. Lin, B.-Y. Wong, C. Liu, and H. Chen. Programmable aperture photography: Multiplexed light field acquisition. *ACM Trans. Graph.*, 27(3):55:1–55:10, 2008. 21

[55] C.K. Liang, G. Liu, and H.H. Chen. Light field acquisition using programmable aperture camera. *IEEE International Conference on Image Processing*, 5:233–236, 2007. 36

[56] A. C. Likas and N. P. Galatsanos. A variational approach for bayesian blind image deconvolution. *IEEE Trans. on Signal Processing*, 52:2222–2233, 2004. 98

[57] C. Liu. Beyond pixels: Exploring new representations and applications for motion analysis. *Doctoral Thesis, Massachusetts Institute of Technology*, May 2009. 110

[58] Li Ma and R.C. Staunton. Integration of multiresolution image segmentation and neural networks for object depth recovery. *Journal of the Pattern Recognition Society*, 38(7):985–996, 2005. 17

[59] M. Martinello and P. Favaro. Fragmented aperture imaging for motion and defocus deblurring. *IEEE International Conference on Image Processing*, Sep 2011. 11

[60] M. Martinello and P. Favaro. Single image blind deconvolution with higher-order texture statistics. In *Video Processing and Computational Video*, volume LNCS7082, pages 124–151. Springer-Verlag, 2011. 10

[61] M. Martinello and P. Favaro. Depth estimation from a video sequence with moving and deformable objects. *Submitted to IET Image Processing*, 2012. 11

[62] M. Martinello, T. E. Bishop, and P. Favaro. A bayesian approach to shape from coded aperture. *IEEE International Conference on Image Processing*, Sep 2010. 10, 130

[63] M. McGuire, W. Matusik, H. Pfister, J. F. Hughes, and F. Durand. Defocus video matting. *ACM Trans. Graph.*, 24(3):567–576, 2005. 20

[64] D.J. McLean. The improvement of images obtained with annular apertures. *Royal Society of London*, 263:545–551, 1961. 54, 80, 130

[65] R. Molina, A. K. Katsaggelos, J. Abad, and J. Mateos. A bayesian approach to blind deconvolution based on dirichlet distributions. *ICASSP*, 4:2809 – 2812, Apr 1997. 98

[66] F. Moreno-Noguer, P. N. Belhumeur, and S. K. Nayar. Active refocusing of images and videos. *ACM Trans. Graph.*, Aug 2007. 18

[67] V. P. Namboodiri and S. Chaudhuri. Recovery of relative depth from a single observation using an uncalibrated (real-aperture) camera. *IEEE Conference on Computer Vision and Patter Recognition*, 2008. 17

[68] S. K. Nayar, M. Watanabe, and M. Noguchi. Real-time focus range sensor. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(12):1186–1198, 1996. 18

[69] R. Ng, M. Levoy, M. Brédif, G. Duval, M. Horowitz, and P. Hanrahan. Light field photography with a hand-held plenoptic camera. Technical Report CSTR 2005-02, Stanford University CS, Apr 2005. 19

[70] D. Nister, H. Stewenius, R. Yang, L. Wang, and Q. Yang. Stereo matching with color-weighted correlation, hierachical belief propagation and occlusion han-

dling. *IEEE Conference on Computer Vision and Patter Recognition*, 2:2347–2354, 2006. 110

[71] B.A. Olshausen and D.J. Field. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381:607–609, 1996. 50

[72] A. P. Pentland. A new sense for depth of field. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(4):523–531, Jul 1987. 26, 30

[73] T. Pouli, D. W. Cunningham, and E. Reinhard. Image statistics and their applications in computer graphics. *Eurographics, State of the Art Report*, May 2010. 72

[74] P. Premaratne and C. Ko. Zero sheet separation of blurred images with symmetrical point spread functions. *Signals, Systems, and Computers*, pages 1297–1299, 1999. 127

[75] W.H. Press, B.P. Flannery, S.A. Teukolsky, and W.T. Vetterling. *Numerical Recipes in C*. Cambridge University Press, 1988. 79

[76] Team AMSL Racing. Meiji university. Nov 2011. URL `http://www.isc.meiji.ac.jp/~amslab/racing/`. 2

[77] A.N.J. Raj and R.C. Staunton. Rational filters design for depth from defocus. *Pattern Recognition*, 45(1):198–207, 2012. 16

[78] A. Rajagopalan and S. Chaudhuri. A variational approach to recovering depth from defocused images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(10):1158–1164, Oct 1997. 31

[79] A. Rajagopalan and S. Chaudhuri. An mrf model-based approach to simultaneous recovery of depth and restoration from defocused images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(7):577–589, Jul 1999. 15, 16, 31

[80] D. L. Ruderman. The statistics of natural images. *Network: Computation in Neural Systems*, 5:517–548, Jul 1994. 72

[81] L. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D*, 60:259–268, 1992. 73

[82] J. Salmon and Y. Strozecki. From patches to pixel in non-local methods: weighted-average reprojection. *IEEE International Conference on Image Processing*, Sep 2010. 111

[83] A. Saxena, S. H. Chung, and A. Y. Ng. Learning depth from single monocular images. *Neural Information Processing Systems*, 18, 2005. 18

[84] A. Saxena, S. H. Chung, and A. Y. Ng. 3-d depth reconstruction from a single still image. *International Journal of Computer Vision*, 76(2):157–173, 2007. 18

[85] Y. Y. Schechner and N. Kiryati. Depth from defocus vs. stereo: How different really are they? *International Journal of Computer Vision*, 39(2):141–162, Sep 2000. 30

[86] Q. Shan, W. Xiong, and J. Jia. Rotational motion deblurring of a rigid object from a single image. *International Conference on Computer Vision*, pages 1–8, Oct 2007. 98

[87] B. Smith, L. Zhang, and H. Jin. Stereo matching with non-parametric smoothness priors in feature space. *IEEE Conference on Computer Vision and Patter Recognition*, pages 485–492, 2009. 110

[88] W. J. Smith. *Modern Optical Engineering*. McGraw-Hill, 2000. 31

[89] D. Snyder, T. Schulz, and J. O'Sullivan. Deblurring subject to nonnegativity constraints. *IEEE Trans. on Signal Processing*, 40(5):1143–1150, 1992. 80

[90] D. Sun, S. Roth, and M. J. Black. Secrets of optical flow estimation and their principles. *IEEE Conference on Computer Vision and Patter Recognition*, Jun 2010. 110

[91] X. Sun and Q. Cheng. On subspace distance. *Image Analysis and Recognition*, pages 81–89, 2006. 125

[92] H. Tao, H. Sawhney, and R. Kumar. Dynamic depth recovery from multiple synchronized video streams. *IEEE Conference on Computer Vision and Patter Recognition*, 1:118–124, 2001. 110

[93] Wikipedia the free encyclopedia. Autostereogram. Feb 2012. URL `http://en.wikipedia.org/wiki/Autostereogram`. 6, 7

[94] C. Tomasi and R. Manduchi. Bilateral filters for gray and color images. *International Conference on Computer Vision*, pages 839–846, 1998. 111

[95] L. Torresani and A. Hertzmann. Automatic non-rigid 3d modeling from video. *European Conference on Computer Vision*, pages 299–312, 2004. 110

[96] L. Torresani, A. Hertzmann, and C. Bregler. Non-rigid structure-from-motion: Estimating shape and motion with hierarchical priors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(5):878–892, May 2008. 110

[97] E. Trucco and A. Verri. *Introductory Techniques for 3-D Computer Vision*. Prentice Hall, 1998. 25

[98] A. Veeraraghavan, R. Raskar, A.K. Agrawal, A. Mohan, and J. Tumblin. Dappled photography: mask enhanced cameras for heterodyned light fields and coded aperture refocusing. *ACM Trans. Graph.*, 26(3):69, Aug 2007. 21, 49, 50, 51, 54, 55, 80, 98, 130

[99] Microsoft Corp. Redmond WA. *KINECT - Method and system for object reconstruction*. 2005. URL `http://www.wipo.int/patentscope/search/en/WO2007043036`. 2

[100] M. Watanabe and S. K. Nayar. Rational filters for passive depth from defocus. *International Journal of Computer Vision*, 27(3):203–225, 1998. 16, 30

[101] O. Whyte, J. Sivic, A. Zisserman, and J. Ponce. Non-uniform deblurring for shaken images. *IEEE Conference on Computer Vision and Patter Recognition*, pages 491–498, Jun 2010. 98

[102] L. Xu and J. Jia. Two-phase kernel estimation for robust motion deblurring. *European Conference on Computer Vision*, Oct 2010. 98

[103] D. M. Young. *Iterative solution of large linear systems*. Academic Press, 1971. 120

[104] G. Zhang, J. Jia, T.-T. Wong, and H. Bao. Consistent depth maps recovery from a video sequence. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 974–988, 2009. 110

[105] G. Zhang, J. Jia, W. Hua, and H. Bao. Robust bilayer segmentation and motion/depth estimation with a handheld camera. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 603–617, 2011. 110

[106] C. Zhou and S. K. Nayar. What are good apertures for defocus deblurring? *IEEE International Conference on Computational Photography*, Apr 2009. 21, 54, 80, 87, 130

[107] C. Zhou, S. Lin, and S. K. Nayar. Coded aperture pairs for depth from defocus. *International Conference on Computer Vision*, Oct 2009. 21

[108] S. Zhu, L. Zhang, and B. M. Smith. Model evolution: An incremental approach to non-rigid structure from motion. *IEEE Conference on Computer Vision and Patter Recognition*, pages 1165–1172, 2010. 110

[109] S. Zhuo and T. Sim. On the recovery of depth from a single defocused image. *CAIP*, pages 889–897, 2009. 17

[110] A. Zomet and S. K. Nayar. Lensless imaging with a controllable aperture. *IEEE Conference on Computer Vision and Patter Recognition*, 1:339–346, 2006. 54